Name : Choong Beng Wei
Matrix No : WEK020038
Course Code : WXES 3182
Thesis Title : Program Advisory System

Abstract

The Academic Advisory System is an advisory system that uses the case based – reasoning technique in advising school leavers to courses that they should pursue in universities.

The main objective of the system is to provide the advisory service of a career counsellor by recommending school leavers who want to further their tertiary education courses that are more suitable for them, based on their academic results and their personality.

The system will only be able to be deployed in Malaysia as the education systems are different in each country, thus making this system not suitable to be used in other countries. The main focus group of the system would be school leavers, either from matriculation or from pre – university courses like Form Six.

The system will be a web based system to facilitate the use of the system anywhere in the world and at the same time providing the advisory service at anytime of the day, unlike a human career counsellor.

The case based reasoning technique is one of the reasoning methods in artificial intelligence where a problem is solved using similar solved cases in the past, by adapting the similar cases to the current problem. This technique closely mimics a human's ability to solve problems, rather than the conventional machine way which is more static.

Acknowledgement

The Program Advisory System is a proposed project for the WXES 3181 and WXES 3182 subjects. In completing this project that lasted for several months, many people have given their unconditional help in making this system a success.

Therefore, I would like to express my gratitude and up most appreciation to these wonderful people:

- Mrs. Norisma Idris, for being a wonderful supervisor for my project and advising me though out the duration of the project development
- Mrs. Siti Soraya Abdul Rahman, for her fair evaluation and kind pointers and advise
- Friends and fellow course mates, for their kind support and constant help that made the project even better.

Another warm thank you for all that has taken part in on way or another, though the development of this project.

Choong Beng Wei

8th March 2005

Table of Contents

List of Tables

List of Figures

List of Equations

Chapter 1: Introduction

Chapter 1: Introduction

1.1 Problem Definition

In the every growing community, each parent would want to have their children to become someone successful in the future. Therefore, there would definitely be extra burden and pressure when students choose their next big step in their life, which is what course to pursue for their tertiary education.

Usually, school leavers may opt to choose the most popular courses like medical, engineering and law. But not all students fit into the mould, and it will eventually prove costly in the end. Judging based solely on the academic qualification alone is not enough to prove that the courses chosen are the right and suitable choices for the students.

Moreover, undertaking one course requires the understanding of the certain skills and talents that is needed to excel the field chosen. Some course like law may require good communication skills and a good memory while others may rely on certain thinking skills and creativity.

Having really understanding a suitable course to undertake may require a student to meet career councelors to enable them to know more on the field that are more preferable than the rest. But unfortunately most students are shy and ignorant of such revenue.

The Program Advisory System would help eliminate such problems by recommending and advising students to undertake courses that are suited to their personality and desires. This is done by the case based reasoning technique, where the problems are matched with similar solved cases stored in the system's memory to come out with a possible solution.

1.2 Objectives of the system

The objectives of the systems are:

- To provide a platform for students to get advise and recommendation on what courses that are more suitable to them
- To eliminate time wastage and cost of having to change courses or profession in the future by guiding students to choose course that they would feel comfortable with in the near future.
- To provide the services of a career counsellor online that is accessible anywhere in the world, with the help of the ever growing Internet services

1.3 Project Scope

The target group intended for this system would be students who intend to further their tertiary education, either straight after STPM examinations or from matriculation. The domain expert used in this system would be a career counsellor that usually provides advises and information about professions and their requirements and skills needed. This system would mimic a career counsellor in the way they analyze ones personality and desires and conclude with a list of recommended field to undertake.

1.4 System Limitation

The system is designed to assist and to advise the user in the courses that deemed to suit the overall personality of the user itself. The recommended solutions are not finalized decisions as it may vary with the options that the user has already have in mind. The academic result will only determine the possible choices to undertake while the personality and desire result will further filter those possible choices to pin point the most recommended courses.

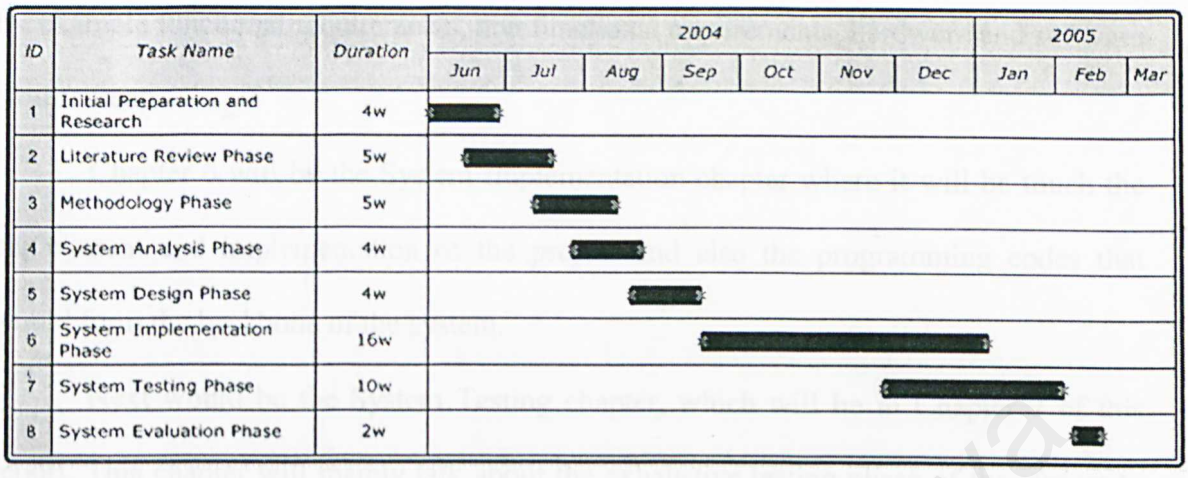
The system also could only cover certain list of professions as it would take a longer period of time and information to actually have a complete database or knowledge base on the professions available in Malaysia or in the world.

1.5 Project Schedule

There are a few phase in developing the Program Advisory System project, which will cover a duration of 10 months. These phases are:

- Literature Review Phase
- Methodology Phase
- System Analysis Phase
- System Design Phase
- System Implementation Phase
- System Testing Phase
- System Evaluation Phase

The project timeline is as of below:



ID	Task Name	Duration	2004							2005		
			Jun	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar
1	Initial Preparation and Research	4w										
2	Literature Review Phase	5w										
3	Methodology Phase	5w										
4	System Analysis Phase	4w										
5	System Design Phase	4w										
6	System Implementation Phase	16w										
7	System Testing Phase	10w										
8	System Evaluation Phase	2w										

Figure 1.1: Project Timeline for the Program Advisory System

1.6 Report Layout

This report is divided into a few parts, all according to the main phases in the project development.

Chapter 1 is the introductory chapter that would explain about the definition of the project, objectives, scope and limitation. Chapter 2 would talk about the literature review that would be the basis of the project that includes the analysis of the information gathering about the project domain and the technical implantation of the project.

Next, the methodology will be in Chapter 3 of this report and it will talk about the technologies and ideas that would be implemented into the project.

Chapter 4 and Chapter 5 are the System Analysis and the System Design chapters respectively. The System Design chapter will talk about the implementation of the design in the project and the

System Analysis chapter would be about the system needs and the requirements, for example functional requirements, non functional requirements, hardware and software requirements.

Chapter 6 will be the System Implementation chapter where it will be touch the deployment and implementation of the project and also the programming codes that would form the backbone of the system.

Next would be the System Testing chapter, which will be in Chapter 7 of this report. This chapter will mainly talk about the exhaustive testing phase of the system to check for errors and bugs.

Last but not to forget, the Chapter 8 would talk about the System Evaluation and Conclusion.

1.7 Chapter Summary

As a summary, this introductory chapter talks briefly about the Program Advisory System by introducing the problem domain, the system objectives and also the overview of the project documentation.

In this chapter also, the limitation of the system is also highlighted and the timeline estimated for the whole duration of the project development is also shown.

In the next chapter, intensive literature review will be done regarding the application of the case based reasoning, technologies used and also the existing system review that has a similarity with the Program Advisory System.

Chapter 2: Literature Review

2.1 Artificial Intelligence
2.2 Expert System: Introduction and Application
2.3 Case Based Reasoning
2.4 Differences between Rule Based Reasoning and Case Based Reasoning
2.5 Technology Review
2.6 Existing System Reviews
2.7 Chapter Summary

Chapter 2: Literature Review

2.1 Artificial Intelligence

2.1.1 Introduction to Artificial Intelligence

The word artificial, from the Merriam Webster Dictionary is defined as 'humanly contrived often on a natural model' while intelligence is 'the ability to learn or understand or to deal with new or trying situations'. Put it together and you will form artificial intelligence, which is the capability of a machine to imitate intelligent human behaviour and it is a branch of computer science dealing with the simulation of intelligent behaviour in computers.

"Artificial Intelligence may be defined as the branch of computer science that is concerned with the automation of intelligent behaviour" (Luger, 2002). So the basic idea of artificial intelligence evolves around imitating the intelligence of a human being and put to it to work into machines.

Artificial intelligence can be divided into certain well known applications, such as stated below:

- Games
- Expert Systems
- Natural language
- Neural Networks
- Robotics

In games, artificial intelligence is used to plot out the possible steps taken in a game of chess or checkers while in expert systems, artificial intelligence is used to provide reasonable and recommended opinion on decision making problems, imitating an expert's judgment and decision making skills and techniques in a particular field. Natural language helps the computers to 'understand' the human language while the neural networks are systems that mimic the physical connections and functions of a human brain.

Last but not least, the robotics fields see the artificial intelligence application in mimicking the human's ability to observe, feel and react though certain stimulus.

2.2 Expert System: Introduction and Application

Expert systems are systems that actually mimic an expert's ability to solve specific problems in their respective domains, thus providing certain problem solving skills that are non – existent in the theoretical knowledge world. These problems solving skills of an expert are derived from past experiences and judgements of similar problems faced by the expert him/herself.

The expert system consist three major components: a knowledge base, a working memory and an inference engine.

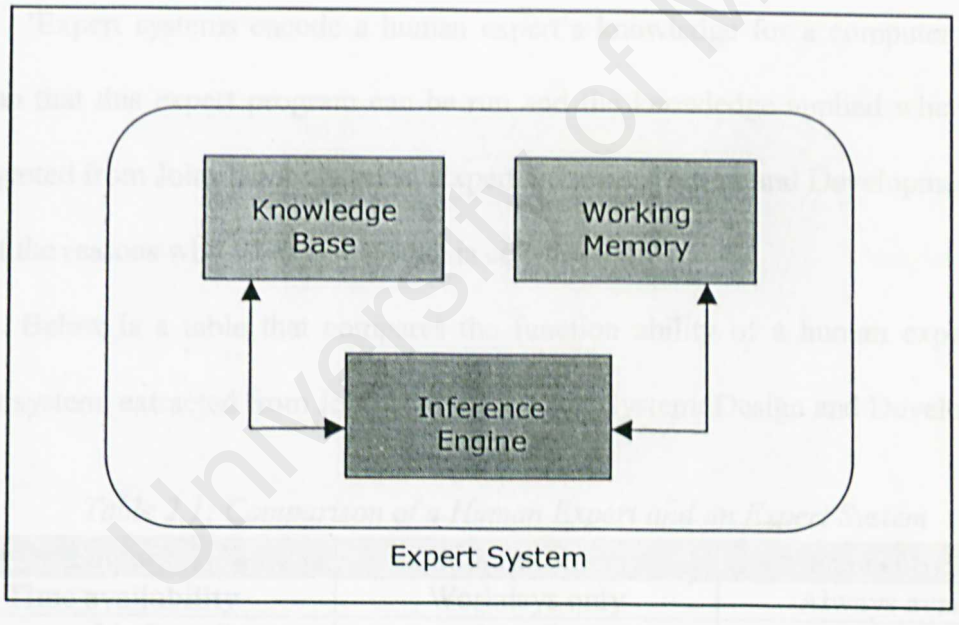


Figure 2.1: Basic Expert System Breakdown

The knowledge base holds knowledge of an expert on a specific field. The knowledge provided by the expert may be in the form of problem facts, rules, concepts and relationships of the domain field.

The inference engine of the expert system would be what we call the knowledge processor, which will be backbone of the system itself. With this inference engine, the expert system would be able to reason like a human expert. This information processor will take into account the information provided by the user on a specific problem or question and it will analyze the information provided with the knowledge stored in the knowledge base. It will then provide the user with reasonable conclusions or recommended answers and solutions to the mentioned problem.

Finally, the working memory functions as a storage area that contains the problem facts that are obtained during the course of analyzing and producing the conclusion or recommended solutions.

'Expert systems encode a human expert's knowledge for a computer in such a fashion that this expert program can be run and the knowledge applied where needed' was quoted from John Durkin's book, Expert Systems: Design and Development. This is one of the reasons why an expert system is constructed.

Below is a table that compares the function ability of a human expert and an expert system, extracted from John Durkin's Expert System: Design and Development.

Table 2.1: Comparison of a Human Expert and an Expert System

Factors	Human Expert	Expert System
Time availability	Workdays only	Always available
Geographic Location	Locally	Anywhere needed
Safety Concerns	Irreplaceable	Replaceable
Perishable	Yes	No
Performance	Varies	Consistent
Speed	Varies	Consistent and faster
Cost	High	Affordable

2.3 Case Based Reasoning

Case based reasoning is based upon the general idea that similar problems faced are best solved using similar solutions encountered in the past thus the important factor in case based reasoning is to learn from experience.

A special characteristic of case based reasoning that is worth mentioning is that this technique of reasoning doesn't require an exact copy of the problem solved in the past to find the solution. Whereas other artificial intelligence techniques that are model based, cased based reasoning excels well when in situations where the problem domain is not well understood by the system or user.

The basic idea of the functionality of a case based reasoning system is that thought retrieval of similar cases from a repository of past cases and adapting it into the current problem domain to come out with a solution. Below depicts the advantages case based reasoning technique hold compared to other artificial intelligence techniques:

- Avoids problems related to the knowledge elicitation and codification
- The need to address only the problems that actually happens while other systems take all possible problems in consideration
- Able to handle failed cases, where it will be able to identify potentially high risk problems
- Able to handle domains that are poorly understood since it contradicts with hypothetical models.
- Solutions recommended are more humanly inclined as it is similar with the human reasoning technique.

Theoretically, there are four important stages of case based reasoning. It is also commonly referred to the R^4 model. They are:

- Retrieve – Retrieve similar cases to the target problem
- Reuse – Reuse past solutions
- Revise – Revise or adapt the suggested solutions to better fit the target problem
- Retain – Retain the target and solution in the case – base (repository).

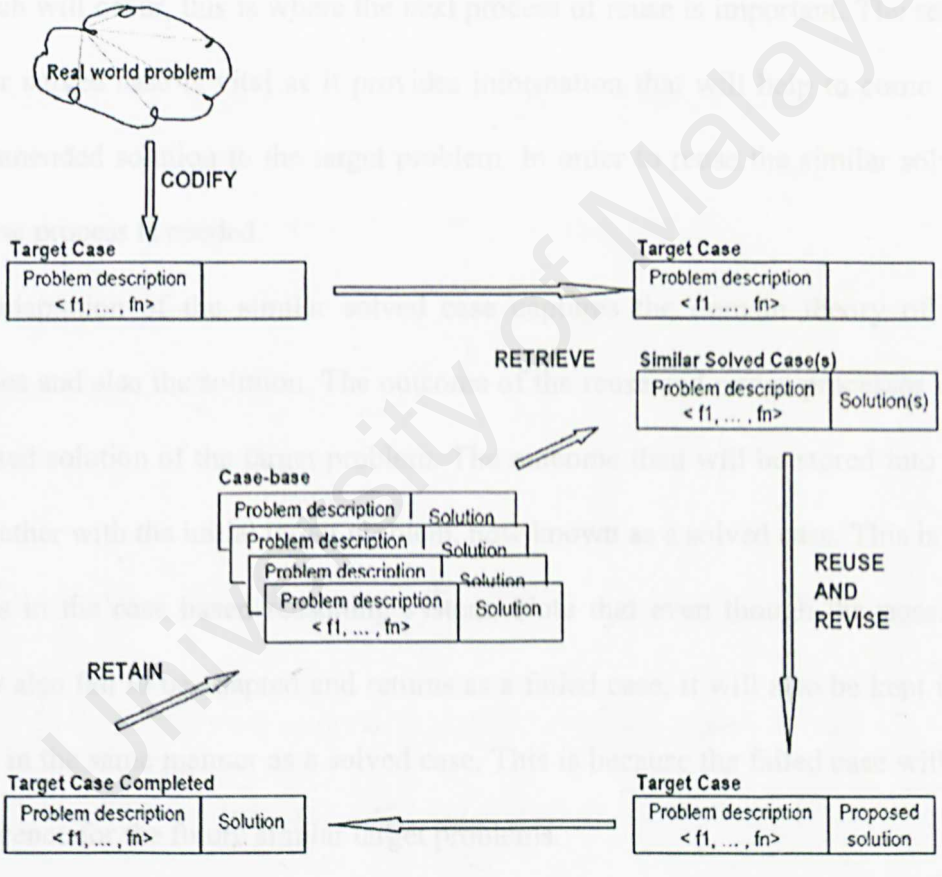


Figure 2.2: Case Based Reasoning Processes (Shephard, unknown)

From the figure, we can see clearly the flow of the case based reasoning process. First and foremost, the real world problem is codified into the system as the target case. This process collects case attributes values that identify the problem type and differentiate one problem type from another. Theses case attributes are used as indices for case storage and retrieval.

The next process is the retrieval process, where the target case is used to find the similar solved case in the repository or case – base of the system. Although it is unlikely an exact match will occur, this is where the next process of reuse is important. The reuse of the similar solved case is vital as it provides information that will help to come out with a recommended solution to the target problem. In order to reuse the similar solved case, the revise process is needed.

The adaptation of the similar solved case captures the domain theory of the attribute values and also the solution. The outcome of the reuse and revise processes will be the proposed solution of the target problem. The outcome then will be stored into the case base together with the initial target problem, now known as a solved case. This is the retain process in the case based reasoning system. Note that even though the possible outcome may also fail to be adapted and returns as a failed case, it will also be kept into the case base in the same manner as a solved case. This is because the failed case will be used as a reference for the future similar target problems.

There are a range of different methods to measure similarity between cases in a case based reasoning system. These similarity method will be vital in deciding the features that are used to rate the similarity between the target problem and the solved cases in the case base. These may include:

- **Nearest Neighbour Algorithms**

  The most popular similarity measure and are based upon straight – forward distance measures for each feature. All features must be standardized to enable an accurate measuring and also to ensure that the choice of unit has no particular influence on the calculation.

  A common algorithm used in the calculation is as of below:

$$SIM(C_1, C_2, P) = \frac{1}{\sqrt{\sum_{j \in p} Feature\_dissimilarity(C_{1j}, C_{2j})}}$$

Equation 2.1: Feature Dissimilarity Formula (Shephard, unknown)

where P is the set of *n* features, C_1 and C_2 are cases and

$$Feature\_dissimilarity(C_{1j}, C_{2j}) \begin{cases} (C_{1j} - C_{2j})^2 \\ 0 \\ 1 \end{cases}$$

Equation 2.2: Feature Dissimilarity Function (Shephard, unknown)

where (i) the features are numeric, (ii) is the feature are categorical and $C_{1j} = C_{2j}$ or (iii) where the features are categorical and $C_{1j} \neq C_{2j}$ respectively.

- **Manually Guided Induction**

  An expert will manually identify the important features, but this is not appropriate as it is clearly not ideal for an expert system where an expert is needed.

- **Template Retrieval**

  This mimics the database retrieval process where the user provides the values for a subset of the problem description and all the cases that matched these values are retrieved.

- **Specificity Preferences**

  Cases are chosen based on the idea of the cases that matched the features exactly rather than those that matched generally.

- **Frequency Preferences**

  Cases that are most frequently retrieved in the past are preferred compared than the rest.

- **Recency Preferences**

  The more recently matched cases are preferred over those that have not been matched for some period of time.

- **Object – Oriented Similarity**

  This algorithm is designed specifically for complex problem domains where there are the necessities to compare between different structured cases.

- **Fuzzy Similarity**

  This method uses the concepts such as the 'at-least-as-similar' and the 'just-noticeable-difference' as opposed to crisp values.

However, these similarity measures too bear their own down sides. They could be a problem when it comes to symbolic or categorical features and also the inability to take into consideration information that can be derived from the structure of the data.

Even though it seems like the case based reasoning is the ultimate technique that similarly mimic the human reasoning ability, it has also its own shortcomings. For instance, if the case based reasoning case – base doesn't have a sufficient similar case to the target problem, the reuse and adapt processes may be inappropriate. This is modelled to a human, when a person encounters a problem that he / she have no previous experience to. Lack of similar cases may tend to make the so – called solution to be bias.

There is also the possibility of the system where it is unable to recognize the new problem type. That is when a new case is distinguished from previous cases by a feature or features are not represented in the indices, the case based reasoning will be unable to recognize the distinction.

The problem of the system's validation process also poses another disadvantage of the case based reasoning. The extent of the limit of the case base data on a specific domain must be taken into account. This is to ensure the system understands and know its own limitations and to determine when the case base data has insufficient similar cases to be used and adapted into the target problem.

2.4 Differences between Rule Based Reasoning and Case Based Reasoning

Rule based expert system is defined as a program that processes problem specific information contained in the working memory with rules stored in the knowledge base using the inference engine to produce new information or perhaps recommendations to the user.

The architecture of a rule based system is similar to the original model of the expert system, but with an addition of a few more subsystems that will complement the system. These subsystems are:

- User Interface
- Developer Interface
- Explanation Facility
- External Programs

In a rule based reasoning system, the rules in the knowledge base is representing the information contained in the long – term memory and the facts contained in the working memory represent the situations in the short term memory. The other major component, the inference engine functions as the reasoning component where it compares the facts with the premises of the rules in the knowledge base to see which rule fits the situation.

The rules that fits or fires in the terms of artificial intelligence, have their own conclusions and these conclusions will be stored in the working memory and the whole process will be iterated until no other rules has anything that matches the facts contained in the working memory.

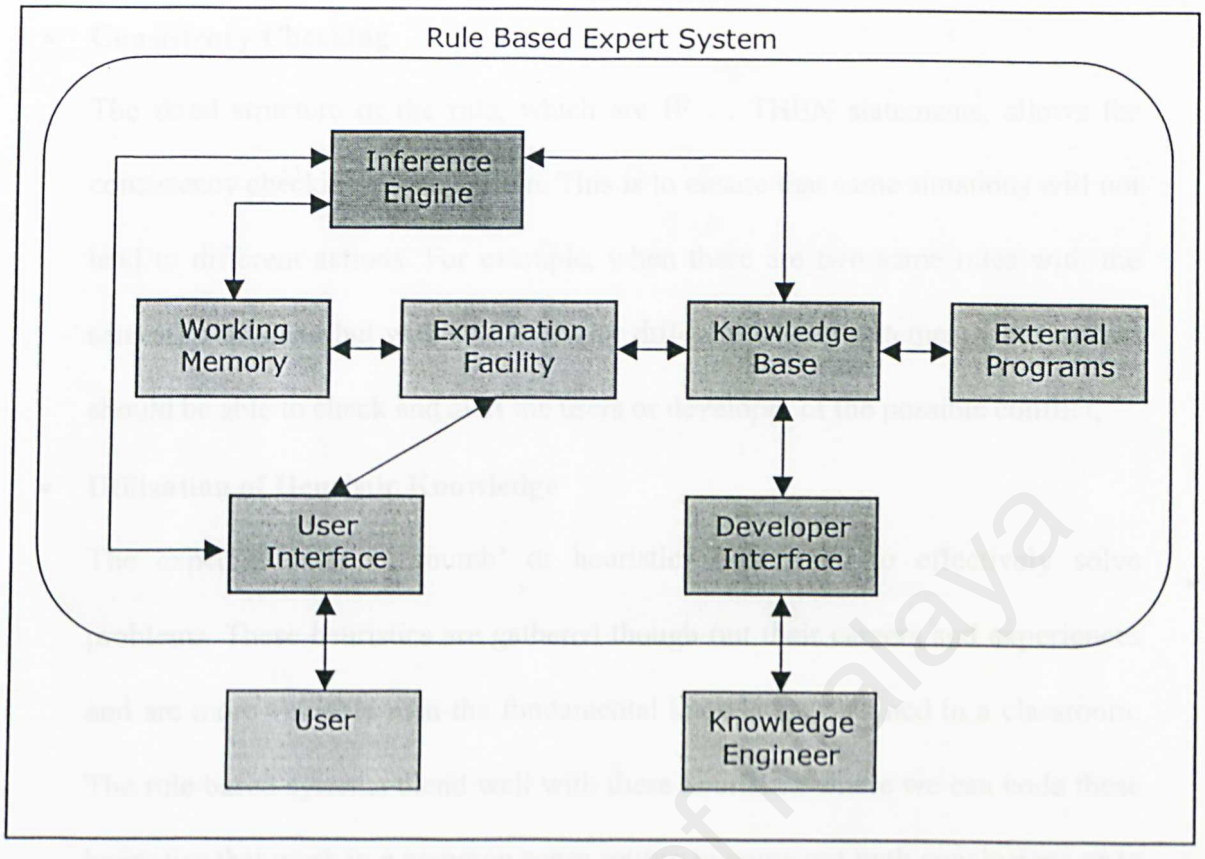


Figure 2.3: Rule Based Expert System Breakdown

There are some distinct advantages of a rule based system, mainly:

- **Use of Relevant Knowledge**

  The system will only use the relevant rules to the problem, where the system will decide on which rules to pursue based on the discovered information in order to solve the problem. This is due because the system may have many rules that can address a number of problem issues.

- **Consistency Checking**

  The fixed structure of the rule, which are IF … THEN statements, allows for consistency checking of the system. This is to ensure that same situations will not lead to different actions. For example, when there are two same rules with the same IF statement but with conflicting or different THEN statements, the system should be able to check and alert the users or developer of the possible conflict,

- **Utilization of Heuristic Knowledge**

  The expert's 'rules of thumb' or heuristics help them to effectively solve problems. These heuristics are gathered though out their careers and experiences and are more valuable then the fundamental knowledge obtained in a classroom. The rule based systems blend well with these heuristics where we can code these heuristics that work in a common sense setting to come out with conclusions or to effectively control the search of the knowledge base.

- **Can Incorporate Variables**

  Variables can be used in the rules where it will further enhance the performance of the rule based system. This also promoted the reuse of rules where it can be applied across similar objects.

- **Modularity of Knowledge**

  A rule itself is an independent piece of information in the knowledge base. It depicts the facts that may be used to solve a problem from the THEN statement by using the IF statements as a comparison. Being independent mean that the rules can be reviewed on its own in order to verify its correctness.

However, these are also some drawbacks of the rule based reasoning systems, where:

- **Require Exact Matching**

  The rule based system matches the facts in the working memory with the rules stored in the knowledge base, meaning that exact matching is required to happen before any rule can be fired and fitted into the working memory for further progress. For instance:

  IF **The coffee is hot**
  THEN **Cool it first**

  If in the working memory we place these two statements to be matched:

  The coffee is piping hot
  The coffee's temperature is hot,

  the system may not have an exact match and the rule will not fire. Even though the meanings of the two statements are similar, as long as the system couldn't match them word by word, the rule will still not fire.

- **Slow Processing Rate**

  To be able to match the rules with the facts in the working memory, the system must do an exhaustive search though out the knowledge base for the exact rule to be fired. Knowing that the knowledge base has a vast amount of rules stored, this will result a slower processing time

- **Have Opaque Rule Relationships**

  It would be difficult to determine how rules are logically related to one another through an inference chain. For example, to have Rule A to fire, Rule B must be met and to have Rule C to fire, Rule C must be fired first. This can go on to a big number of rules before the initial rule to be fired. Moreover, rules can be place

anywhere in a knowledge base and with a large number of rules to match, it will be difficult to locate and trace the correct related rules.

Both case based reasoning systems and rule based reasoning systems have their own distinct advantages and disadvantages. Below would be a comparison table between these two major reasoning methods of artificial intelligence study.

Table 2.2: Comparison of case based reasoning systems with rule based reasoning systems

Criteria	Case Based Reasoning	Rule Based Reasoning
Situations Used	Poorly understood problem area with complex structured data that changes slowly with time and justification required	Well-understood, stable, narrow problem area and justification by rule-trace acceptable
Knowledge Unit	Cases	Rules
Processing Time	Longer	Shorter
Learning Ability	No	Yes
Explanation mechanism	Backtrack of rule firings	Similar solved cases
Growth	Easy as cases are individual units of it own	Difficult as rules need to be rewritten
Solution Obtained	Based on the rules fired	Based on adaptation of similar cases
Advantages	Easy to set up a knowledge base. Cases matched based on similarity between problem and solved cases in the case base.	Ability to incorporate variables into rules. Consistency checking for possible problematic rules in the knowledge base
Disadvantages	Similarity measuring method not fully tested. May not be able to recognize a new case.	Requires exact matching of rules to enable them to be fired. Difficult to capture knowledge on a problem domain in a set of rules

2.5 Technology Review

2.5.1 System Architectures

2.5.1.1 Microsoft .Net Framework

The .NET Framework is a new computing platform that simplifies application development in the highly distributed environment of the Internet. The .NET Framework is designed to fulfil the following objectives:

- To provide a consistent object-oriented programming environment whether object code is stored and executed locally, executed locally but distributed through the Internet, or executed remotely.
- To provide a code-execution environment that minimizes software deployment and versioning conflicts.
- To provide a code-execution environment that guarantees safe execution of code, including code created by an unknown or semi-trusted third party.
- To provide a code-execution environment that eliminates the performance problems of scripted or interpreted environments.
- To make the developer experience consistent across widely varying types of applications, such as Windows-based applications and Web-based applications.
- To build all communication on industry standards to ensure that code based on the .NET Framework can integrate with any other code.

The .NET Framework has two main components: the common language runtime and the .NET Framework class library. The common language runtime is the foundation of the .NET Framework where the runtime can be regarded as an agent that manages code at execution time, providing core services such as memory management, thread

management, and remoting, while also enforcing strict type safety and other forms of code accuracy. In fact, the concept of code management is a fundamental principle of the runtime. Code that targets the runtime is known as managed code, while code that does not target the runtime is known as unmanaged code. The class library, the other main component of the .NET Framework, is a comprehensive, object-oriented collection of reusable types that you can use to develop applications ranging from traditional command-line or graphical user interface (GUI) applications to applications based on the innovations provided by ASP.NET, such as Web Forms and XML Web services.

Features of the Common Language Runtime

The common language runtime manages memory, thread execution, code execution, code safety verification, compilation, and other system services. These features are intrinsic to the managed code that runs on the common language runtime.

With regards to security, managed components are awarded varying degrees of trust, depending on a number of factors that include their origin (such as the Internet, enterprise network, or local computer). This means that a managed component might or might not be able to perform file-access operations, registry-access operations, or other sensitive functions, even if it is being used in the same active application.

The runtime enforces code access security. For example, users can trust that an executable embedded in a Web page can play an animation on screen or sing a song, but cannot access their personal data, file system, or network

The runtime also enforces code robustness by implementing a strict type-and-code-verification infrastructure called the common type system (CTS). The CTS ensures that all managed code is self-describing. The various Microsoft and third-party language

compilers generate managed code that conforms to the CTS. This means that managed code can consume other managed types and instances, while strictly enforcing type fidelity and type safety.

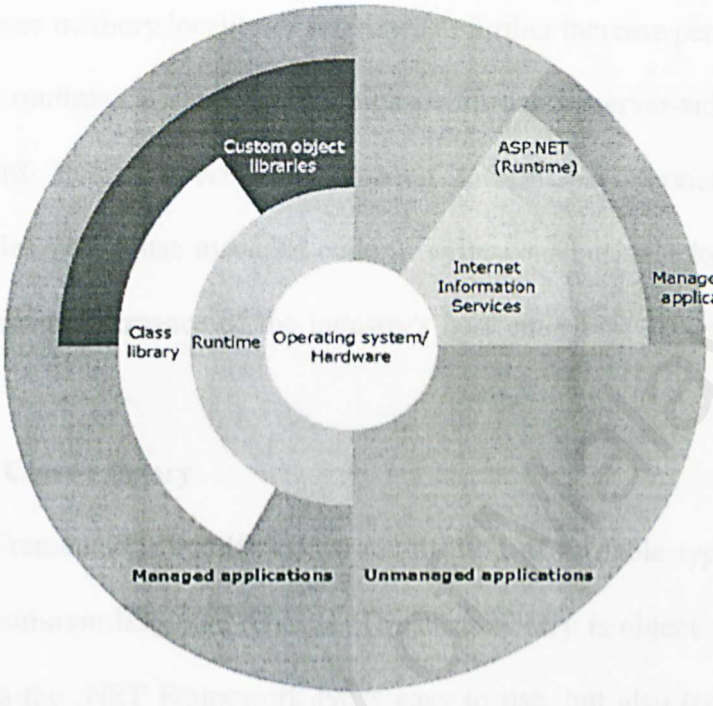


Figure 2.4: Relationship of the common language runtime and the class library to the applications and to the overall system.

In addition, the managed environment of the runtime eliminates many common software issues. The runtime also accelerates developer productivity. For example, programmers can write applications in their development language of choice, yet take full advantage of the runtime, the class library, and components written in other languages by other developers.

While the runtime is designed for the software of the future, it also supports software of today and yesterday. The runtime is designed to enhance performance. Although the common language runtime provides many standard runtime services,

managed code is never interpreted. A feature called just-in-time (JIT) compiling enables all managed code to run in the native machine language of the system on which it is executing. Meanwhile, the memory manager removes the possibilities of fragmented memory and increases memory locality-of-reference to further increase performance.

Finally, the runtime can be hosted by high-performance, server-side applications, such as Microsoft® SQL Server™ and Internet Information Services (IIS). This infrastructure enables you to use managed code to write your business logic, while still enjoying the superior performance of the industry's best enterprise servers that support runtime hosting.

.NET Framework Class Library

The .NET Framework class library is a collection of reusable types that tightly integrate with the common language runtime. The class library is object - oriented, and this not only makes the .NET Framework types easy to use, but also reduces the time associated with learning new features of the .NET Framework. In addition, third-party components can integrate without great hassle with classes in the .NET Framework.

As developers would expect from an object-oriented class library, the .NET Framework types enable them to accomplish a range of common programming tasks, including tasks such as string management, data collection, database connectivity, and file access. In addition to these common tasks, the class library includes types that support a variety of specialized development scenarios. For example, developers can use the .NET Framework to develop the following types of applications and services:

- Console applications.
- Windows GUI applications (Windows Forms).

- ASP.NET applications.
- XML Web services.
- Windows services.

Client Application Development

Client applications are the closest to a traditional style of application in Windows-based programming. These are the types of applications that display windows or forms on the desktop, enabling a user to perform a task. Client applications include applications such as word processors and spreadsheets, as well as custom business applications such as data-entry tools, reporting tools, and so on. Client applications usually employ windows, menus, buttons, and other GUI elements, and they likely access local resources such as the file system and peripherals such as printers.

For instance, the Windows Forms classes contained in the .NET Framework are designed to be used for GUI development. Developers easily can create command windows, buttons, menus, toolbars, and other screen elements with the flexibility necessary to accommodate shifting business needs.

Server Application Development

Server-side applications in the managed world are implemented through runtime hosts. Unmanaged applications host the common language runtime, which allows custom managed code to control the behaviour of the server. This model provides users with all the features of the common language runtime and class library while gaining the performance and scalability of the host server.

Server-side managed code

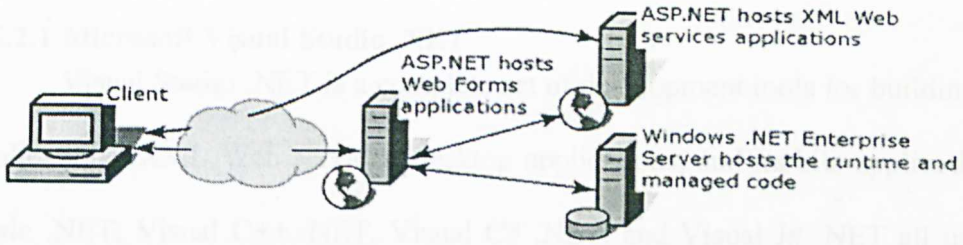


Figure 2.5: A basic network schema with managed code running in different server environments.

ASP.NET is the hosting environment that enables developers to use the .NET Framework to target Web-based applications. However, ASP.NET is more than just a runtime host; it is a complete architecture for developing Web sites and Internet-distributed objects using managed code. Both Web Forms and XML Web services use IIS and ASP.NET as the publishing mechanism for applications, and both have a collection of supporting classes in the .NET Framework.

XML Web services, an important evolution in Web-based technology, are distributed, server-side application components similar to common Web sites. However, unlike Web-based applications, XML Web services components have no UI and are not targeted for browsers such as Internet Explorer and Netscape Navigator. Instead, XML Web services consist of reusable software components designed to be used by other applications, such as traditional client applications, Web-based applications, or even other XML Web services. As a result, XML Web services technology is rapidly moving application development and deployment into the highly distributed environment of the Internet.

2.5.2 Programming Languages

2.5.2.1 Microsoft Visual Studio .NET

Visual Studio .NET is a complete set of development tools for building ASP Web applications, XML Web services, desktop applications, and mobile applications. Visual Basic .NET, Visual C++ .NET, Visual C# .NET, and Visual J# .NET all use the same integrated development environment (IDE), which allows them to share tools and facilitates in the creation of mixed-language solutions. In addition, these languages blend in with the functionality of the .NET Framework, which provides access to key technologies that simplify the development of ASP Web applications and XML Web services.

Visual J#

Visual J# is a development tool that developers who are familiar with the Java-language syntax can use to build applications and services on the .NET Framework. It integrates the Java-language syntax into the Visual Studio .NET integrated development environment (IDE). Visual J# is not a tool for developing applications intended to run on a Java Virtual Machine. Applications and services built with Visual J# will run only in the .NET Framework. Visual J# has been independently developed by Microsoft and it is not endorsed or approved by Sun Microsystems, Inc.

Smart Device Applications

The Visual Studio .NET integrated development environment now includes tools for developing applications for smart devices, such as the Pocket PC. Using the tools and the .NET Compact Framework users can create, build, debug, and deploy applications that run on the .NET Compact Framework in personal digital assistants (PDAs), mobile phones, and other resource-constrained devices.

ASP.NET Mobile Designer

ASP.NET Mobile Designer extends ASP.NET and the .NET Framework, allowing you to build Web applications for mobile phones, Personal Digital Assistants (PDA), and pagers. This designer is integrated into the Visual Studio IDE. Developers can create mobile Web applications, use the Mobile Designer to modify a mobile Web form, and then build and run the application, all from within Visual Studio.

Web Forms

Web Forms are an ASP.NET technology that is used to create programmable Web pages. Web Forms render themselves as browser-compatible HTML and script, which allows any browser on any platform to view the pages. Using Web Forms, you create Web pages by dragging and dropping controls onto the designer and then adding code, similar to the way that is used to create Visual Basic forms.

Windows Forms

Windows Forms is the new platform for Microsoft Windows application development, based on the .NET Framework. This framework provides a clear, object-oriented, extensible set of classes that enables you to develop rich Windows applications.

XML Web Services

XML Web services are applications that can receive requests and data using XML over HTTP. XML Web services are not tied to a particular component technology or object-calling convention and can therefore be accessed by any language, component model, or operating system. In Visual Studio .NET, you can quickly create and include XML Web services using Visual Basic, Visual C#, JScript, Managed Extensions for C++, or ATL Server.

XML Support

Extensible Markup Language (XML) provides a method for describing structured data. XML is a subset of SGML that is optimized for delivery over the Web. The World Wide Web Consortium (W3C) defines XML standards so that structured data will be uniform and independent of applications. Visual Studio .NET fully supports XML, providing the XML Designer to make it easier to edit XML and create XML schemas.

2.5.2.2 JavaScript

JavaScript is a very well known scripting language used by many in the world of programming. It is commonly used to improve the web pages design by add interactivity to HTML pages, validate forms and more.

A scripting language is defined as a lightweight programming language and functions nicely when it is embedded directly in HTML pages. Most users do no need to register or purchase a license before using JavaScript and were developed by Netscape and basically work well in all major browsers.

Not forgetting, JavaScript is an interpreted language, meaning that scripts execute without preliminary compilation of the codes.

Below are some of the benefits of JavaScript:

- JavaScript gives HTML designers a programming tool as it is a scripting language with a very simple syntax. It is a relatively easy to insert small lines of code into HTML pages to further enhance its capabilities.
- JavaScript can put dynamic text into an HTML page by using statement like:

  document.write("<h1>" + name + "</h1>")

  which can write a variable text into an HTML page.
- JavaScript can react to events by executing commands or codes when something happens, like when a page has finished loading.
- JavaScript can read and write HTML elements
- JavaScript can be used to validate data before it is submitted to a server as this will save the server from extra processing

2.5.3 Databases

2.5.3.1 Microsoft SQL Server

Microsoft SQL Server is a relational database management system (RDBMS) which is part of Microsoft's BackOffice family of servers. SQL Server was designed for client/server use and is accessed by applications using SQL. It runs on Windows NT version 3.5 or higher and it is compliant with the ANSI SQL – 92 and FIPS 127 – 2 SQL standards.

SQL Server supports symmetric multiprocessing hardware, Simple Network Management Protocol or SNMP, Open Database Connectivity or ODBC and major open standard communications protocols. It has Internet integration, data replication, data warehousing features and also reporting services available.

Microsoft SQL Server was originally developed by Sybase Corporation but the cooperation was broken before the version 6.0.

Currently, the latest version of SQL Server is the SQL Server 2000 and the upcoming version would be SQL Server 2005 that boasts a wide range of upgrades and capabilities.

SQL Server provides a better and more powerful tool for databases compare to the other version of database management system by Microsoft, which is Microsoft Access. Both cater different kinds of data types and variables.

2.6 Existing System Reviews

2.6.1 MyMajors – www.mymajors.com

MyMajor is advisory system for people who want to further their studies to colleges and universities. This system will advise and recommend college majors and universities majors based on the level of education the user has reached, basically catering only for high school senior and college freshmen in the United States.

This system is online based and has been around for since 2003. It interface is based on Active Server Pages and has database connectivity to store users information.

These are some of the MyMajors advisory system's functionalities:

- **User Registration**

  This is one feature that is provided by the system, where a new user has to register him / she self before they can use the system. This is to facilitate the system to retrieve the gathered information from the user and store it into the system database so that when the user logs in again next time, the system can 'recognize' the user and the information supplied by the user during the previous visit.

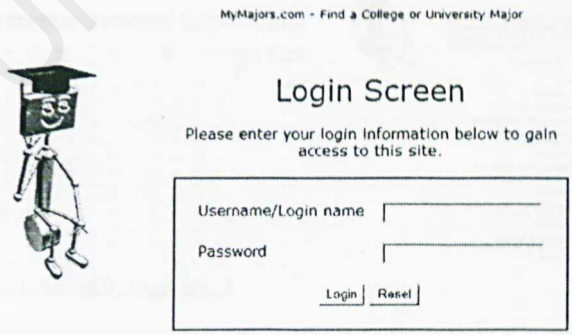


Figure 2.6 Login Screen of the MyMajors advisory system

- **Grade Level and Exam Results Recognition**

  The next functionally featured in the system is the grade level and exam results recognition, where the user will need to input the current level of education and the results of the major exams taken.

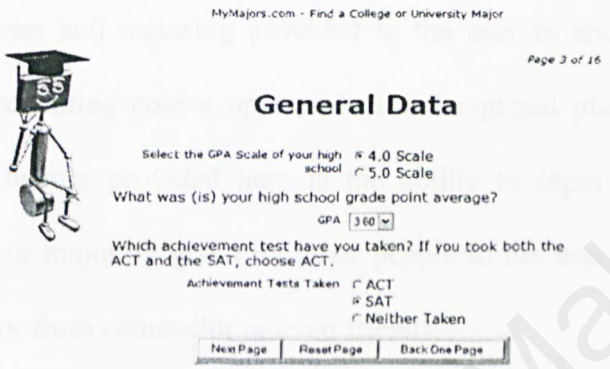


Figure 2.7 User GPA and Exam Level Acquisition Screen

- **Course Achievement and Enjoyment**

  The course achievement and enjoyment function is to gather the user's grade according to the list of courses given and the level of enjoyment each individual course has given to them, the user.

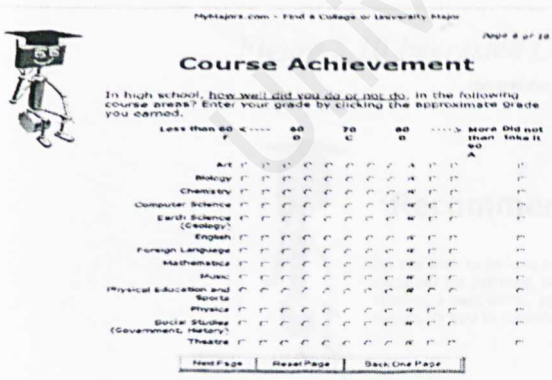


Figure 2.8 Course Achievement Screen

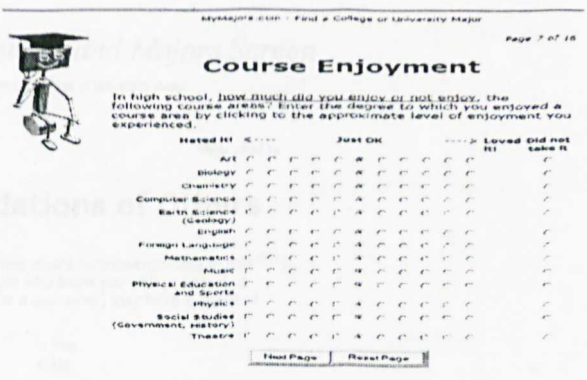


Figure 2.9 Course Enjoyment Screen

- **Interested Courses and Majors**

The system also provided a list of courses for the user to choose from in order to determine their more preferred courses or majoring to pursue. At the same time, the system will also prompt the user to select the most unlikely courses or majoring that the user will not pursue at all. There are 60 courses and majoring provided to the user to choose from, ranging from accounting course up to urban and regional planning course. One unique feature provided here is the ability to input the recommended courses or majoring given by other people to the user, for instance from parents or from counsellor or even friends.

Figure 2.10 Interested Courses and Majors Screen

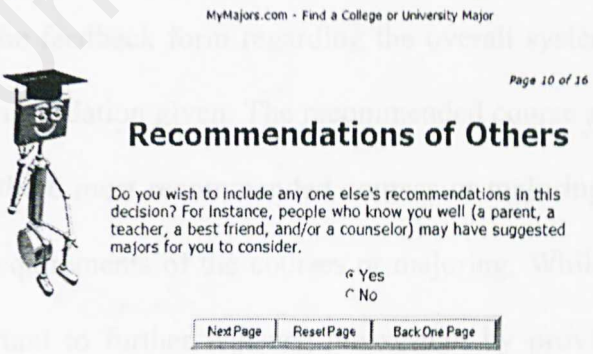


Figure 2.11 Recommendations of Others Screen

- **Preferences and Values Questionnaire**

  Almost at the end of the information acquisition of the system, the user is prompted to answer a questionnaire regarding their preferences and values on certain issues or questions. These will finally be included during the analysis of the courses and majoring recommended by the system to the user. It is more like a personality test and a judgement of likes and dislikes of the user.

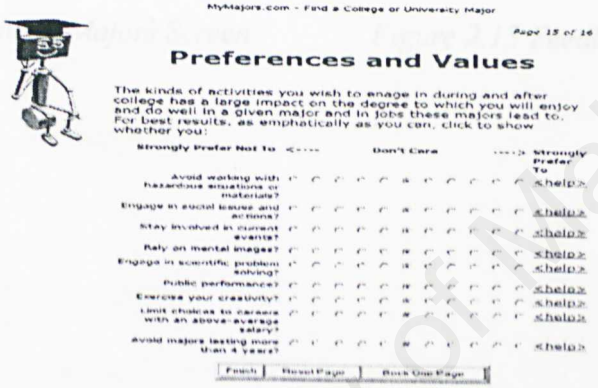


Figure 2.11 Preferences and Values Questionnaire Screen

- **Recommended Course and Feedback**

  The final functionality feature is the display of the recommended results and the feedback form regarding the overall system performance and the recommendation given. The recommended course screen provides the user with the 6 most recommended courses or majoring and also their outline and requirements of the courses or majoring. While, the feedback form is important to further enhance the system by providing useful comments and feedback from the user to the system developers and owners.

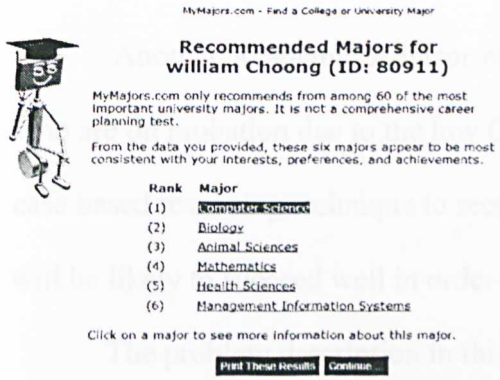


Figure 2.12 Recommended Majors Screen

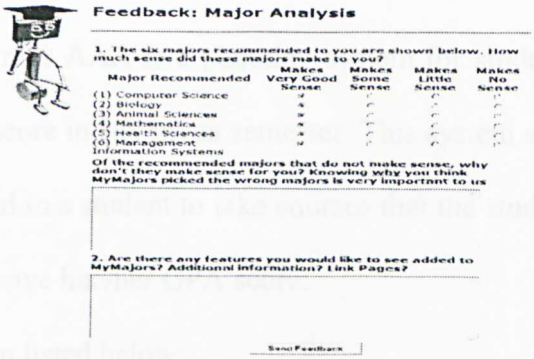


Figure 2.13 Feedback Form Screen

2.6.2 Another Academic Advisor

Another Academic Advisor or known as AAA is a planning system for students who are on probation due to the low CGPA score in any given semester. This system uses case based reasoning technique to recommend to a student to take courses that the student will be likely to succeed well in order to improve his /her GPA score.

The problem description in this system listed below:

- Gender
- Ethnic Group
- Current Standing
- Major
- Accumulative GPA
- High School Rank Percent
- High School GPA
- List of taken Course

The system will be using the Nearest Neighbour algorithm for case retrieval as the cases will be stored in a flat file. The numeric value that will be calculated and used for the Nearest Neighbour algorithm will be the accumulative GPA, high school GPA and the rank percent in high school.

$$Sim(x, y) : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \rightarrow [0, 1]$$

$$Sim(x, y) = \begin{cases} \frac{\min(x,y)}{\max(x,y)}, & \text{if } max(x, y) \neq 0 \\ 0, & \text{otherwise} \end{cases}$$

Equation 2.3 Similarity Calculation between two values (Binh,unknown)

The system flow is depicted as below:

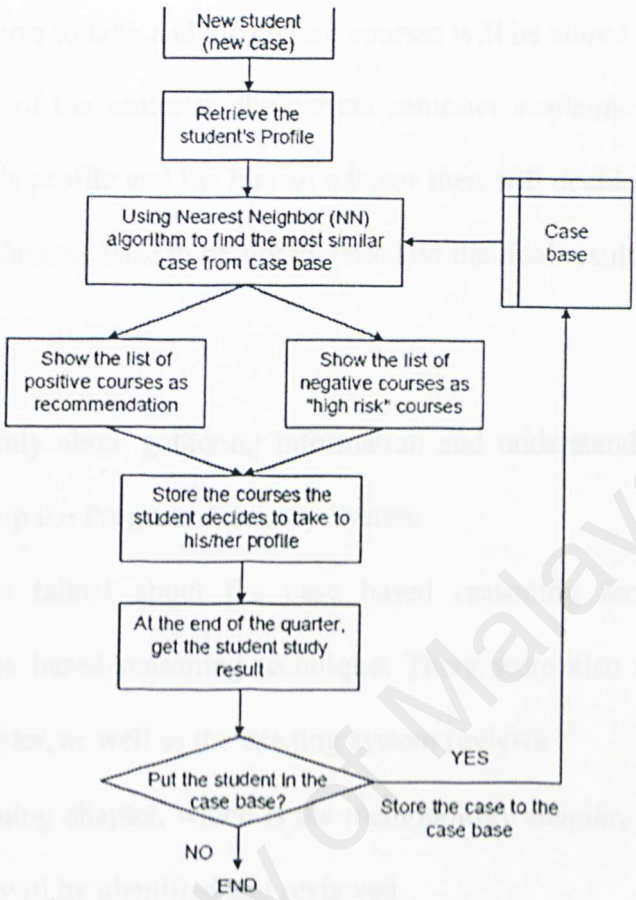


Figure 2.14 AAA System Flow (Binh, unknown)

The system will first prompt the user to key in the student's profile which consists of the student's academic and personal information. Then the AAA system will use the Nearest Neighbour algorithm to find the most similar case solved in the repository or known as case base. The case retrieved will have the list of courses divided into the positive and negative course groups, where the courses in the positive course group will be recommended to the student while the courses in the negative course group will be highlighted as the high – risk courses to take.

The student then will decide on what courses he / she should take based on the recommended list of course to take and the chosen courses will be stored in the student's own profile. At the end of the semester, the current semester academic results will be updated into the student's profile and the human advisor then will decide whether to put the updated profile into the case base to be stored, based on the final result of the profile.

2.7 Chapter Summary

Chapter 2 is mainly about gathering information and understanding the system domain in order to develop the Program Advisory System.

This chapter has talked about the case based reasoning technique and it comparison with the rule based reasoning technique. There were also the technology reviews made in this chapter, as well as the existing system reviews.

In the next upcoming chapter, which is the methodology chapter, a few different software process models will be identified and reviewed.

Chapter 3: Methodology

Chapter 3: Methodology

3.1 Software Development Life – Cycle Model

3.1.1 Introduction

Software development life – cycle model (SDLC) or also known as software process model is defined as a conceptual model which is used in project management where it describes the phases needed to develop a software system.

The phases in detail are:

- **Requirement phase**

  This is the initial phase of a project where the client's requirements are taken into account and the concept is explored and refined.

- **Analysis phase**

  The analysis phase is also known as the specification phase as the client's requirements are analyzed and documented in the form of a specification document where it states what the system is supposed to function for. The software project management plan is also prepared in this phase where it depicts the software development in detail.

- **Design phase**

  The specification documentation prepared in the earlier phase is used to design the system. This is where the architectural design and the detail design of the system is done to describe how the system looks and functions.

- **Implementation phase**

  The implementation phase overlooks the process of coding and testing of the system based on the documents prepared in the earlier phases. Acceptance test is also preformed in this phase by the client as soon as the final system has been integrated together and further system testing is done.

- **Maintenance phase**

  Corrective maintenance, perfective maintenance and adaptive maintenance sum up the last phase where at this phase, the final system is deployed and running at the client's place.

3.1.2 Different Kinds of Life - Cycle Models

There are a few different kinds of software life – cycle models, namely Waterfall Model, Iterative and Incremental Model and Rapid – Prototyping Model. These models have their own characteristics and have different benefits and downsides.

Waterfall Model

Waterfall model is a linear life cycle model equipped with feedback loops. These feedback loops enable changes to be made to the specific phases whenever there is a change of requirements.

An important point of this model is that no particular phase is complete when the documentation of that phase has been completed and the deliverables of the phase has been checked by the software quality assurance group, as known as the SQA group.

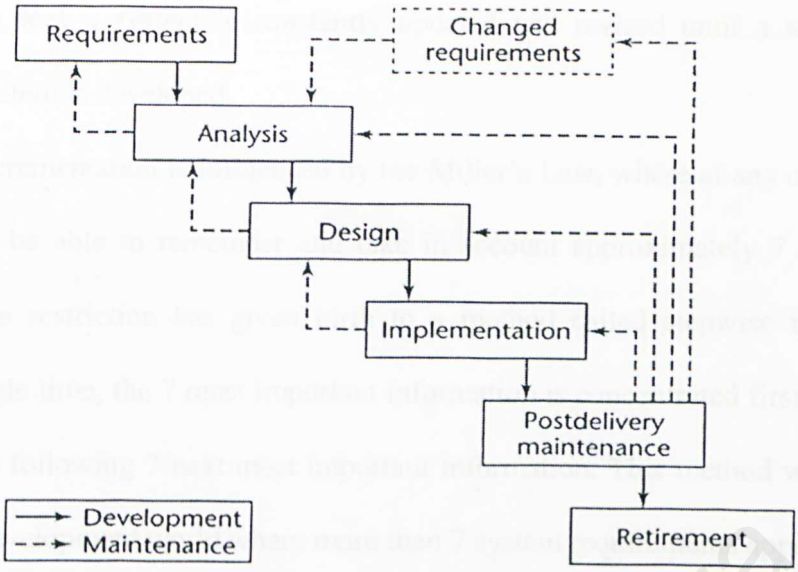


Figure 3.1: Waterfall Life Cycle Model (Schach, 2005)

Testing in this model is an ongoing process and it has no special phase for it. It is also not done only at the end of each phase but through out the whole phase development.

The advantages of the Waterfall model are the enforced disciplined approach where documentations is to be provided at each phase and the products of each phase is to be approved by the SQA group.

However, the downsides of this model are the highly technical documentation prepared and also the fact that the earliest time a client can see the working product is only after all the entire product documentation is done. This may lead to a misunderstanding of the flow and real needs of the clients in the end of the day.

Iterative and Incremental Model

Another well known software life – cycle model is the Iterative and Incremental Model. This model is create as iteration is an important aspect in software development

where iterations sees a system is constantly updated and revised until a satisfactory version of the system is developed.

While incrementation is influenced by the Miller's Law, where at any one time, a human can only be able to remember and take in account approximately 7 chunks of information. This restriction has given birth to a method called stepwise refinement where at one single time, the 7 most important information is concentrated first, and then proceeded by the following 7 next most important information. This method works well in the software development world where more than 7 system requirements appear.

All these factors has made the Iterative and Incremental life cycle model a reality, as what is depict on the figure below:

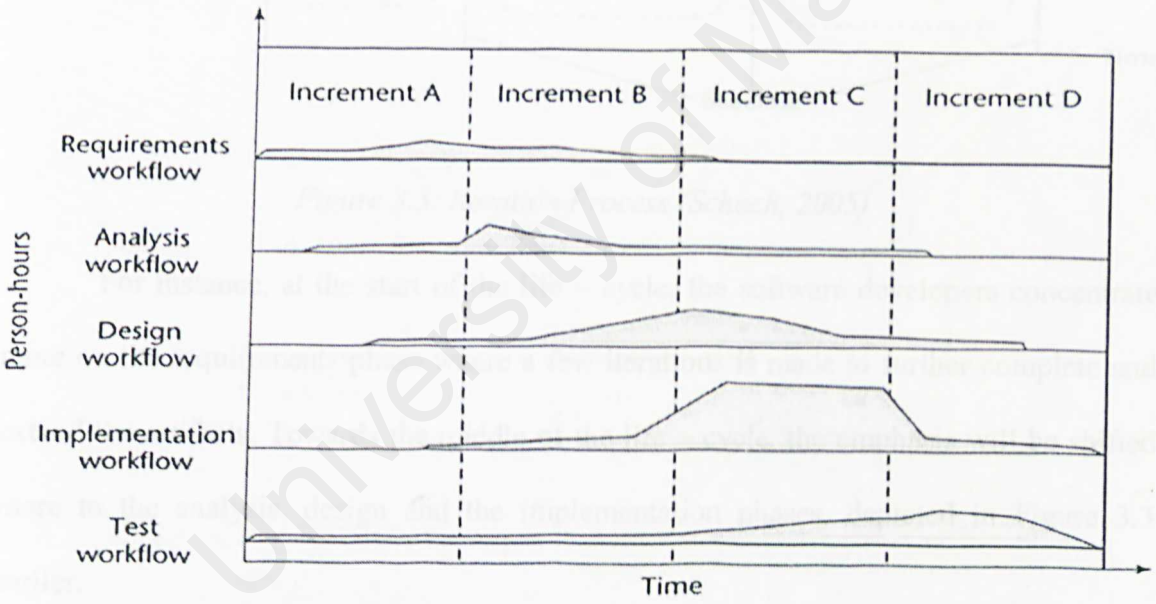


Figure 3.2: Iterative and Incremental Life Cycle Model (Schach, 2005)

As we can see, the 4 major phases of a software development, represented here as workflows are divided into 4 increments. Each increment sees the iteration of the 4 workflows, from the Requirements workflow to the Test workflow. The progress of each

increment at each phase is depicted with the purple shaded area of the curve at each workflow.

The next diagram will clearly show the iteration of the workflow better.

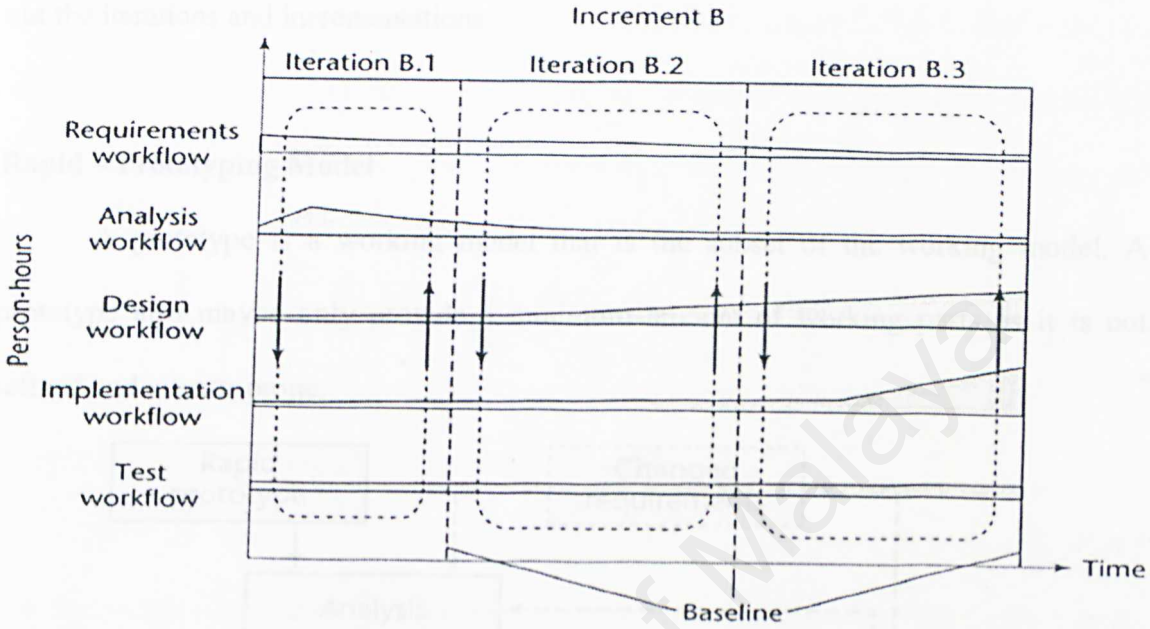


Figure 3.3: Iteration Process (Schach, 2005)

For instance, at the start of the life – cycle, the software developers concentrate more on the requirements phase where a few iterations is made to further complete and extend the artifacts. Towards the middle of the life – cycle, the emphasis will be shifted more to the analysis, design and the implementation phases, depicted in Figure 3.3 earlier.

It is to be note that each incrementation has its own number of iterations, depending on the workload and enhancements done at each incrementation. And also, each of the iteration is not done solely on one specific workflow only. It involves all four workflows, but at certain iterations any one of the workflow may predominate.

At a glance, the Iterative and Incremental life – cycle can be considered as a set of mini projects where each projects are incremented to form the whole complete system. Each mini project iterates between the four major workflows and testing is done through out the iterations and incrementations.

Rapid – Prototyping Model

A prototype is a working model that is the subset of the working model. A prototype also maybe only provide a minimum amount of working parts as it is not refined and is error prone.

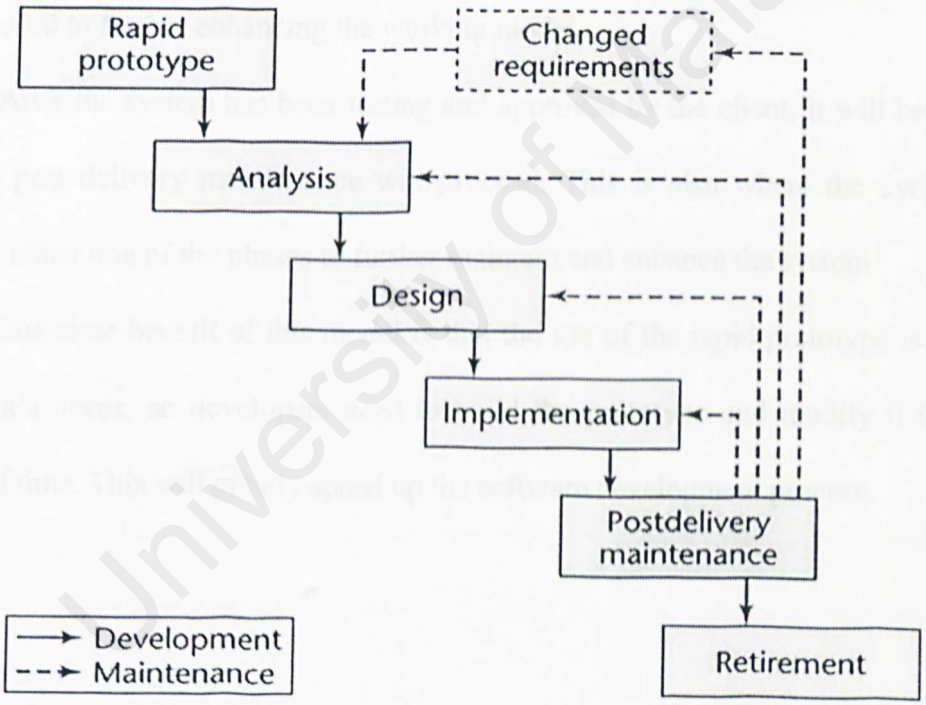


Figure 3.4: Rapid Prototyping Life – Cycle Model (Schach, 2005)

However, the rapid – prototyping life – cycle model's main goal is to build a prototype for it to be view by the clients and other end users for them to interact and to

provide productive feedbacks to the software developers to enable them to construct a specification document that is more or less able to meet the client's real world needs.

One of the major key points of this life – cycle model is the linearity of the model, where feedback loops used in the Waterfall Model can be ignored. This is due to the fact that the prototype is used to construct the specification document, thus making the documentation correct as the prototype some how or rather has been validate by the clients.

The rapid prototyping life – cycle also eases the implementation phase as the initial prototype has given some insights to the developers how the design should be, and just proceed to further enhancing the working model.

After the system has been testing and approved by the client, it will be deployed and the post delivery maintenance will proceed. This is also where the cycle repeats itself, at either one of the phases to further maintain and enhance the system

One clear benefit of this model is that the use of the rapid prototype is to reflect the client's needs, so developers need to build the prototype and modify it in a short period of time. This will greatly speed up the software development process.

3.2 Software Life – Cycle Selected

For the Program Advisory Program, the software life – cycle selected will be the Rapid Prototyping Life – Cycle Model.

There are many benefits that can be said of the life – cycle model selected. Mainly, the reasons for it are:

- **Able to capture the client's and end – user's needs more effectively**

  For the Program Advisory System, the domain users would be career counsellors and future student who want to pursue their tertiary education. To be able to grasp what their needs are may be difficult using the conventional way of interviewing and surveying.

  By rapid prototyping, the domain users can see a glimpse of the upcoming system and it will be easier for them to express their thoughts and ideas through the prototype. In this way, the complete system will be suited to their needs from the very first day itself.

- **Software development processes made easier**

  Through rapid prototyping, the initial prototype that has been built will be discarded. But the initial prototype holds the key points to the design and implementation of the final system.

  This is due to the fact that the system's initial design has been seen by the client and also the end user. Their feedback on the design will be vital in modifying the original design specification rather than to have to change it later during the implementation phase, like the other life – cycle models.

Moreover, the software developers have insights of the final system, all thanks to the rapid prototype that they constructed early in the software development process.

3.3 Research Techniques and Methods

To be able to come out with a relatively good system, initial research and development is vital. The kinds of information perceived during the research period also influence the level of understanding of the domain problems that may arise during the design and implementation process of the system.

A good level of understand the domain knowledge also curb from any misunderstanding of concepts during data acquisition and knowledge engineering.

For the development of the Program Advisory System, the methods used to research and gather information related to the systems are:

- **Interviewing**

  To full understand the way a career counsellor guide an individual to choose a course that is deemed suitable for him / herself, interviews may be the best way to capture the heuristics and experiences used. This valuable knowledge cannot be put to words at times and cannot be found in books or any published papers.

  Interviews will also be done to the other domain users to grasp the ideal outlook of the system and its functionality.

- **Internet**

  The Internet has proven time and time again as a major source of information and knowledge. Tons and tons of information can be gathered through the Internet, and communication between people is far more efficient through the Internet.

- **Reference books and research papers**

  References books and research papers are one of the reliable sources of information to people. These written products are the brainchild of authors who spend many sleepless nights doing intensive research and development to bring and to interpret knowledge into words can last forever.

- **Related thesis done**

  Past thesis done may provide insights of how a thesis should look like and the do's and don't while documenting a thesis. One way or another, we can learn a lot from just reviewing past thesis, either to grab hold of the formats used or maybe to provide vital information where other resources couldn't.

- **Discussions with Supervisors and Moderators**

  Supervisors are the best source for points and reviews, as they are more well - versed in the given field of research. They are the ones that will see the common errors committed and should be a good place to seek more insights of the research area

3.4 Chapter Summary

In Chapter 3, 3 methodologies, namely the Waterfall Life Cycle Model, the Iterative and Incremental Life Cycle Model and the Rapid Prototyping Model were introduced together with their pros and cons.

This chapter also talked about the research techniques and methods used for this project. In the next chapter, Chapter 4 will be about the system analysis where the functional and non – functional requirements are identified.

Chapter 4: System Analysis

Chapter 4: System Analysis

4.1 Introduction

In a software development process, the first and foremost important process that needs to be completed it the requirements engineering process, where most of the system's requirements are identified and taken into consideration

The term requirements in the world of software development can be related to a high – level abstract statement of a service or of a system constraint to a detailed mathematical functional specification. To put into a lay man's words, requirements mean what a person wants, and in this case what the client wants to have in the system that is to be developed.

Requirements engineering is the process of establishing the services that the client wants from a system and the limitations of what the system can do. It is a very important process because wrong requirements may not reflect the real needs of a client in the system and also it would be expensive to make changes after it has been agreed to in the first place.

Basically, requirements are divided into two groups, which are the functional requirements and also the non – functional requirements. The detailed explanation of these requirements will be discussed in the next few sub chapters, together with the Program Advisory System's functional and non – functional requirements.

4.2 Functional Requirements

Functional Requirements is defined as the statements of services the system should provide, how the system will react to situations and the limitations of the system.

The functional system requirements describe the system services in great detail while the functional user requirements are high – level statements of what the system should do and provide.

For the Program Advisory System, the functional requirements are as of below:

- **Information Gathering**

  The system must be able to gather relevant information about the domain user in order to begin the advisory process. The information gathered will be converted into a structured template to be easily reviewed.

  This is done by requesting the needed information in a presentable media, like for instance a form, where the information the user keyed into the form will used.

- **Information Analysis**

  The Program Advisory System main functionality is to be able to perform information analysis to come out with a conclusion to the problem target of the client or the user. This will be done using the case – based reasoning technique where the problem target is compared with the similar solved cases stored in the database.

  The comparison will based on the key features identifies and deemed the important points that can distinguish any given problem domain. The system will

be able to 'think' like a human and give conclusion like any human expert would do.

- **Recommendations and Advices**

  The system should be able to present the analyzed information and produce solutions. Note that the solutions produced shouldn't be considered as 100 percent correct, but would be a recommended answer to the given problem target.

  Suitable advices or supporting ideas should also be given to further support the recommended solution to give some sort of an explanation to the user. This system should be taken as only an advisory source.

Next, use – case modelling is a process of modelling a system's functionality in the shape of business events, by the person who initiates the events and by how will the system responds to those initiated events.

Use case diagram is a diagram that is used to show the interactions between the system and the external systems and users. It depicts the users of the system and how will they be able to interact with the system.

The following figure will show the use case diagram for the Program Advisory System.

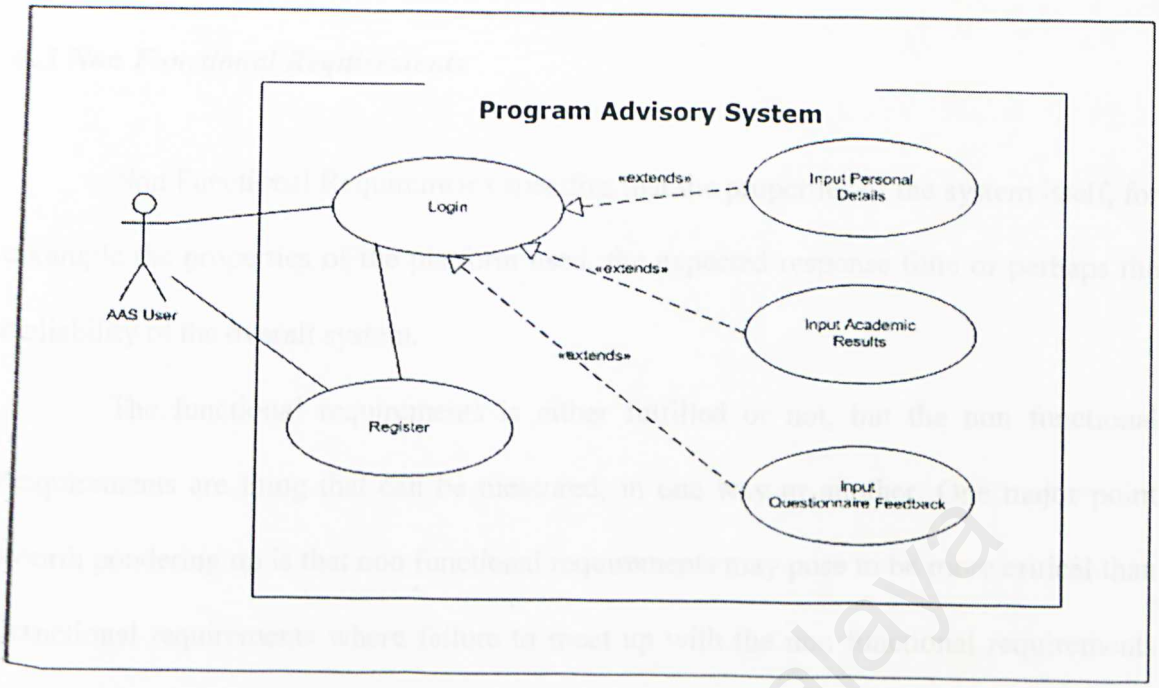


Figure 4.1 Use - Case Diagram for Program Advisory System

4.3 Non Functional Requirements

Non Functional Requirements specifies that the properties of the system itself, for example the properties of the platform used, the expected response time or perhaps the reliability of the overall system.

The functional requirements is either fulfilled or not, but the non functional requirements are thing that can be measured, in one way or another. One major point worth pondering on is that non functional requirements may pose to be more critical than functional requirements where failure to meet up with the non functional requirements may cause the whole system to useless.

The non functional requirements of this system are:

- **Usability**

  This system would be very user –friendly as the interface that is to be integrated will be pleasing to the eye and also with the actions keys and button placed at strategically position areas for easy manoeuvring.

  The system also will be easy to use as most part of the system has easy to read instructions to help the user to go around the system

- **Efficiency**

  The system will be high efficient as every transaction of information will be done at real time. Waiting time will be at a minimal figure and loading of the system will not burden the client's hardware resources.

- **Reliability**

The Program Advisory System will be place in a very reliable platform and this will further enhance the reliability factor. The system will be equipped with error checking facilities and error messages in events of any problems or complications encountered.

- **Portability**

The system is highly portable as it will be deployed in a web server and can be accessed anywhere in the world through the Internet. Just as long as the user has an internet connection, he or she can access this system online.

- **Expandability**

The system should be able to undergo changes to provide newer features, functionality and future enhancement of the database.

4.4 Hardware Requirements

4.4.1 Server Side Hardware Requirements

- Server with a minimum of 1.6 MHz processor speed
- At least 256 MB RAM (preferably 512 MB and above)
- 120 GB HDD space
- Network connection
- UPS system
- Other standard hardware peripherals

4.4.2 Client Side Hardware Requirements

- Personal Computer with a minimum of 300 MHz processor speed
- At least 64 MB RAM
- 10 GB HDD space
- Internet connection with a minimum of 36.6 kBps bandwidth
- Integrated Graphics Card that supports at least 800 x 600 pixel resolution and at least 16 bit colour quality
- Other standard hardware peripherals

4.5 Software Requirements

4.5.1 Server Side Software Requirements

- Microsoft Windows XP Professional with the latest service packs
- Microsoft Internet Information Services
- Microsoft .Net Framework 1.1
- Microsoft SQL Server 2000
- Microsoft Internet Explorer 6.0

4.5.2 Client Side Software Requirements

- Microsoft Windows 98 or higher
- Microsoft .Net Framework 1.1
- Microsoft Internet Explorer 6.0

4.6 Development Tools and Technologies

4.6.1 Operating System / Platform – Windows XP Professional

To develop the Program Advisory System, Windows XP Professional is chosen to be main developing environment. Being developed based on the same foundation that has proved Windows 2000 success, the Windows XP Professional provides the same feature Windows 2000 can provide and even more in fact.

It is proven that programs and task that runs under Windows XP Professional are performing better than ever before and the start up time is way less short. Coupled with enhanced features for the business front and also for advanced users, Windows XP Professional provides more options for software developers every where with more control and reliability.

Update on the security and integrity of the sensitive information in Windows XP Professional has made it even better. Microsoft has also provided a built-in firewall for internet connectivity that make web - surfing more secure. Other than that, with the new Automated System Recovery feature, Windows XP Professional can restore the operating system back in to the original state in just a few clicks away.

Also, Windows XP Professional can be deployed incrementally throughout any organization. Upgrading individual computers only as needed reduces costs.

Business computers linked via Windows 2000 Server software are a cinch to upgrade and manage using Remote Installation Services and Group Policies. These features let you install, configure, and manage individual computers as groups rather than single machines, and monitor them from a centralized location for greater savings in time and support costs.

User customizations also are easier with Windows XP Professional. Different users with different levels of access to the computer have not been easy in the past version of Windows but with Windows XP Professional, more can be done.

4.6.2 Web Server - Internet Information Services (IIS)

Internet Information Services (IIS) is a Microsoft product that comes bundled with Windows XP Professional, like the previous version of Windows.

IIS can transmit information using the Hypertext Transfer Protocol (HTTP) and also can be configured to provide File Transfer Protocol (FTP) and gopher services. The FTP service allows files transfers to and from any FTP sites. The gopher service uses a menu-driven protocol for locating documents. However, gopher protocol has been overshadowed by the newer HTTP protocol.

Besides that, IIS also provide other services like Simple Mail Transfer Protocol (SMTP), Network News Transfer Protocol (NNTP), Microsoft Transaction Server, Microsoft Index Server, Microsoft Certificate Server and Microsoft Management Console (MMC).

Microsoft includes into the IIS a set of programs for developing and administering Web sites, a search engine that allows users to create customized search forms with a variety of tools, including ASP, ActiveX Data Objects, and SQL database queries, reporting tools from Crystal Reports, which is a visual reporting tool that lets the user create presentation-quality reports and integrate them into database applications and also support for writing Web-based applications that can be used to access databases.

4.6.3 Database Server – Microsoft SQL Server 2000

Microsoft SQL Server 2000 all round SQL database that is targeted for the small or midsize organizations. This newer version of the SQL server provides more user – friendly features and also at the same be able to multi task without showing any signs of performance drop.

Microsoft SQL Server 2000 has also a lot of new features that further enhances the control the user has over the data management platform. For example, it supports symmetric multiprocessing hardware, Simple Network Management Protocol or SNMP, Open Database Connectivity or ODBC and major open standard communications protocols and also has Internet integration, data replication, data warehousing features and reporting services available to users, to name a few.

Microsoft SQL Server 2000 is renowned for its scalability and speed and it is also a fully Web-enabled database product that providing support for Extensible Markup Language (XML) and the ability to query across the Internet and beyond the firewall.

4.6.4 Programming Tools - Microsoft Visual Studio .NET

Microsoft Visual Studio .NET is a new age programming tool that provides a platform for multiple programming languages such as C++, VB.NET, C#, ASP.NET, etc. It also provides plug-in interface that allows interaction between third party tools like Macromedia Flash MX.

Next, .NET is connected to database server using the ADO+ or ADO.NET. ADO.NET is the newer form of ADO, which built around n-tier development and architected with XML at its core.

Besides that it also provides XML Web services that can be constructed using ASP.NET. These web services can be accessed through a network connection and data will be primarily sent in XML format. As a result, programs written in any language, using any component model, running on any operating system can access XML Web services and yet understand the message received.

4.6.6 Designing Tools - Adobe Photoshop CS

Adobe Photoshop CS is yet another version of the powerful designer tool that allows user to create great looking pieces of graphic work. Photoshop is image processing tool that can be used to create original work, correct the color dimensions, retouch and refurbish images and prepare high - quality separations and output with more control than before.

Using powerful painting and selection tools, multiple layers, special effects filters, and lighting effects, Adobe Photoshop CS can be a magic wand for creative minds to create graphic images that can be used for almost anything at all.

4.7 Chapter Summary

In this chapter, the functional and non – functional requirement were identified and mentioned. A use case diagram to depict the system activities has been constructed to enable a better understanding of the system functionality.

The development tools and technologies that is to be used in the development of the Program Advisory System were also given an introduction in this chapter. Hardware and software requirements for both the client and server machines were also identified and given consideration of.

Chapter 5, the System Design chapter will next explain about the system's designs and implementation plans as well as the system architectural design that is to be used for the system.

Chapter 5: System Design

Chapter 5: System Design

5.1 Introduction

System design is process where the requirements of the system are converted into a specification to depict what the system will look like and how it is to be done. During the system analysis, the business problems are emphasized while in the system design process, the technical and the implementation will the major concerns.

There are a few approaches to the system design process. They are:

- Model – driven
- Rapid Application Development (RAD)
- Joint Application Development (JAD)

The model – driven approach is a system design approach that concentrates more on drawing system models to document the technical and implementation aspects of a system.

Next, the Rapid Application Development or RAD is an approach that emphasizes on structured, prototyping and JAD techniques to develop a system at a faster rate. It merges the data – driven information engineering, prototyping and JAD techniques into one, making it able to accelerate system development.

While Joint Application Development or JAD is technique that complements other system analysis and design techniques by using participative development among the clients, end – users, system designers and developers to come out with come out and to sort design issue and system deliverables.

For the Program Advisory System, the model – driven approach is used for the system design process.

5.2 System Architecture

For the Program Advisory System, the 3 –tier system architecture design will be used. The application of this system architecture is divided into three tiers, mainly the presentation layer, the application layer or known as the middle layer and the data layer.

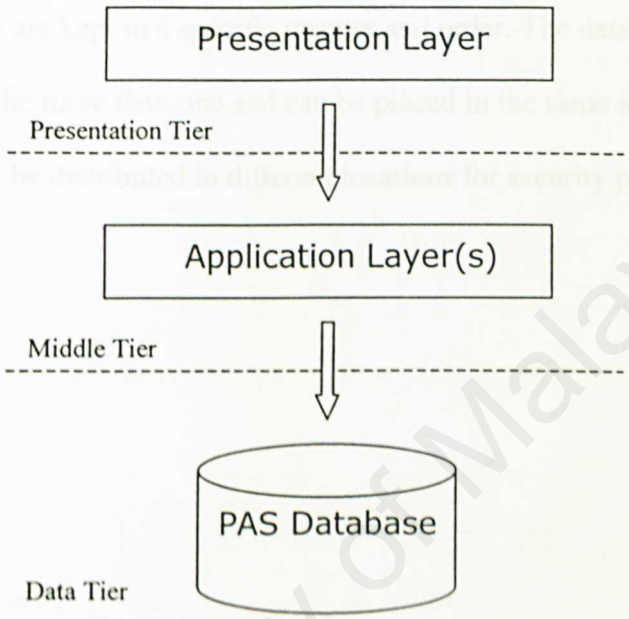


Figure 5.1 Program Advisory System's 3 – Tier System Architecture Design

- **The Presentation Layer**

  The presentation layer is basically graphical user interface layer that function to provide an interface for the end user or the clients to interact with. As most of the business logic is implemented in the middle layer, less computing resources is needed in this layer

- **The Middle Layer**

  This layer is mostly in charge of the business logic and the main processes of the whole system. This layer can be considered as the engine room of the

system where all the system processes are done and is sent to the presentation layer to be displayed to the clients.

- **The Data Layer**

  The data layer is the database layer where all the transaction details and information are kept in a specific manner and order. The database in the data layer can always be more than one and can be placed in the same location as the middle layer or can be distributed to different locations for security purposes.

5.3 Process Design

5.3.1 Functional Decomposition Diagram

The functional decomposition diagram is used to depict the decomposition of a system. This diagram is able to partition the system into smaller subsystems or modules.

The Program Advisory System's functional decomposition diagram is as of below:

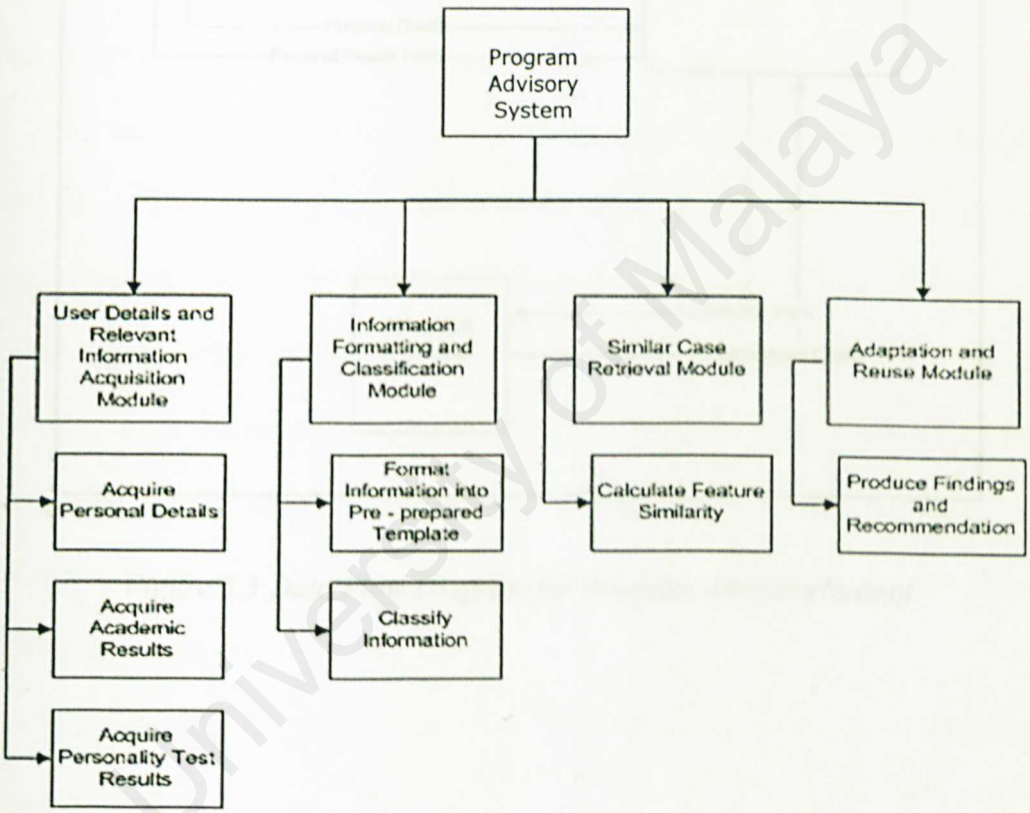


Figure 5.2 Functional Decomposition Diagram for Program Advisory System

5.3.2 Data Flow Diagram

To show the process design of the Program Advisory System, a data flow diagram is created to depict the processes of the system.

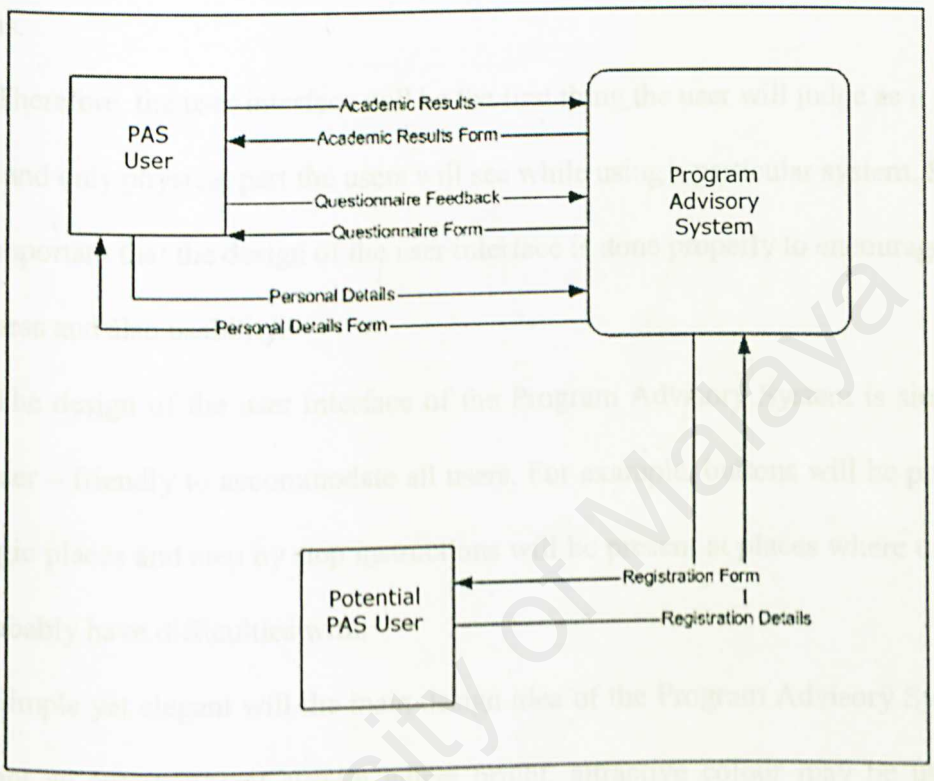


Figure 5.3 Data Flow Diagram for Program Advisory System

5.4 User Interface Design

The user interface is one of the important parts of a system, where it becomes a medium for the user to communicate with the system to perform any transaction or functions.

Therefore, the user interface will be the first thing the user will judge as it will be the first and only physical part the users will see while using a particular system. So, it is vitally important that the design of the user interface is done properly to encourage user – friendliness and also usability.

The design of the user interface of the Program Advisory System is simple and yet be user – friendly to accommodate all users. For example, buttons will be positioned at strategic places and step by step instructions will be present at places where users will most probably have difficulties with.

Simple yet elegant will the main design idea of the Program Advisory System, as this is not an entertainment system where bright, attractive colour may be used. The design of the Program Advisory System may give the users a warm, cooling environment and at the same time user – friendly.

The following figures are the screenshots of the basic design of the Program Advisory System.

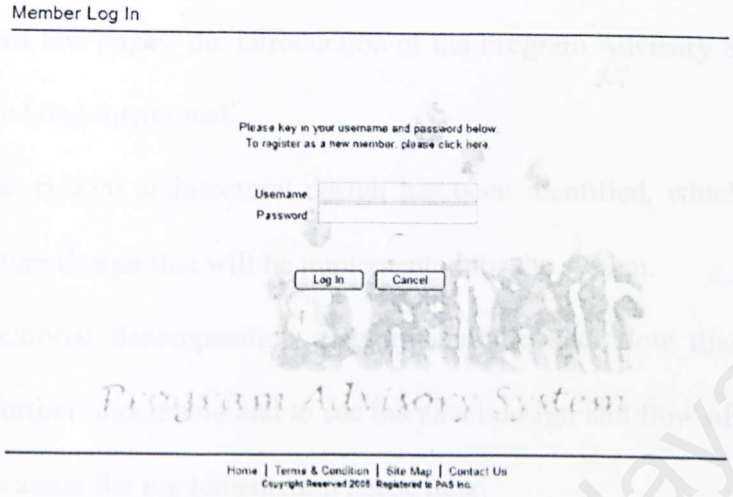


Figure 5.4 Member Registration Screen of Program Advisory System

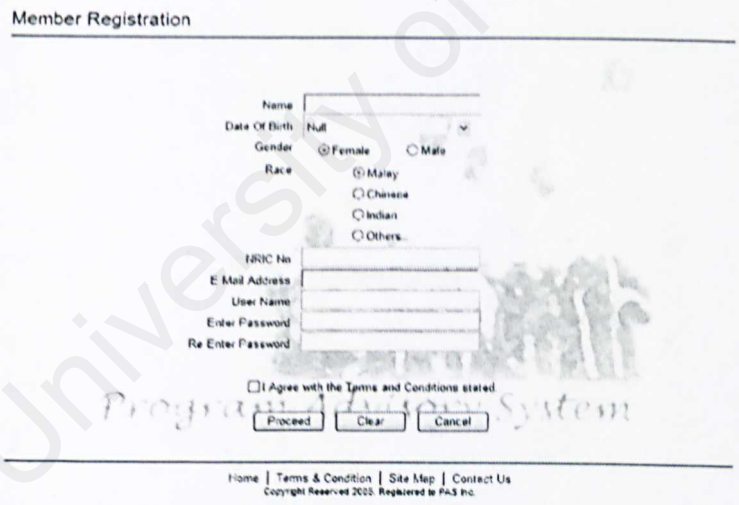


Figure 5.5 Member Login Screen of Program Advisory System

5.5 Chapter Summary

In the last few pages, the introduction of the Program Advisory System's designs has been depicted and mentioned.

Also, the system architectural design has been identified, which was the 3 tier system architecture design that will be implemented for the system.

The functional decomposition diagram and the data flow diagram were also constructed to further understand and to see the exact design and flow of the system, and at the same time assist the implementation phase later.

The next chapter, which is the implementation chapter, will talk about the system development and implementation progress.

Chapter 6: System Implementation

Chapter 6: System Implementation

6.1 Introduction

System Implementation is the process where the planned system, specified during the system analysis and the system design processes is being implemented and developed by system developers. The whole implementation process will oversee the construction of the system, the database needed to support the information and data for the system and also other related services or applications needed to complete the system.

6.2 Development Environment

The development environment of the Program Advisory System is mainly based on the Microsoft .Net technology, where it uses most of the Microsoft components, services and also application and softwares to support its development. Listed below are the lists of software and hardware components used during the development of the Program Advisory System.

6.2.1 Softwares Used for Development

- Windows XP Professional with Service Pack 2
- Microsoft Internet Information Service 6.0 (IIS 6.0)
- Microsoft Internet Explorer 6.0
- Microsoft Outlook Express
- Microsoft Visual Studio .Net 2003
- Microsoft SQL Server 2000 Enterprise Edition
- Microsoft .Net Framework 1.1
- Infragistics NetSuite 2004

- Adobe Photoshop CS
- Adobe ImageReady CS

6.2.1 Hardware Used for Development

- Intel Pentium 3 750Mhz
- 512MB SDRAM 133Mhz
- 80 GB HDD space
- 17in Monitor running on 1152 X 864 pixels resolution
- 32MB NVidia Riva TNT
- Other computer peripherals

6.3 System Development and Implementation

The Program Advisory System is a web based system that would be deployed into the Internet for easy access and to promote mobility around the country. The system is mainly developed using ASP.Net interface, coupled with Visual Basic programming language as the backend programming structure. The system also uses a third party tool, Infragistics NetSuite to further enhance certain interface features that are more attractive and powerful in the same. It uses a SQL Server database to retrieve and store user records and information.

6.3.1 PAS Database Connection

Program Advisory System or PAS connects to a SQL Server database using SQL Server Client which is one of the connection services offered by the Microsoft .Net technology.

The codes used to establish the connection between PAS and the SQL Server database is as of below:

- clsDBConnection In VB Codes

```
Imports System.Data.SqlClient

Public Class clsDBConnection
    'Function: getSQLConnction
    'Description: Connecting to application database
    Function getSQLConnection(ByRef sqlConnection As SqlConnection,
ByRef strErrMsg As String) As Boolean
        Dim strConnection As String =
ConfigurationSettings.AppSettings("DBConnection")
        Dim boolConnection As Boolean
        Try
            sqlConnection.ConnectionString = strConnection
            If sqlConnection.State = ConnectionState.Open Then
                sqlConnection.Close()
            End If
            sqlConnection.Open()

            If sqlConnection.State = ConnectionState.Open Then
                boolConnection = True
            Else
                boolConnection = False
            End If


        Catch ex As Exception
            strErrMsg = ex.Message
            boolConnection = False
        End Try
        Return boolConnection
    End Function

End Class
```

In order to retrieve records from the database, PAS will invoke the codes above to first establish a connection between the web page and the SQL server. Then, it will next invoke another set of codes to specifically request, create or delete information or data in the database. Usually, these codes are store in class files, used in Visual Studio .Net to promote reusability. This is due to the fact that whenever there is the need to invoke codes to retrieve information, we do not need to retype each and single line of codes again in the modules of system, where as of now, we can optimize performance and processing time by just invoking the functions stored in the class file.

- clsMember in VB Codes

```
Imports System.Data.SqlClient
Imports System.Web.Security

Public Class clsMember
    Private clsDBConnection As New clsDBConnection
    Private clsItemId As New clsItemID

    Public Function bindMemberLoginAdapter(ByRef strErrMsg As String,
ByRef scnConnection As SqlConnection, ByRef sdaLogin As SqlDataAdapter,
_
   ByVal strUsername As String)
        Dim blnResult As Boolean
        Dim scmCommand As New SqlCommand

        blnResult = clsDBConnection.getSQLConnection(scnConnection,
strErrMsg)

        If blnResult Then
            Try
                scmCommand = New SqlCommand("spMemberLogin",
scnConnection)
                sdaLogin.SelectCommand = scmCommand
                sdaLogin.SelectCommand.CommandType =
CommandType.StoredProcedure
                scmCommand.Parameters.Add("@Username",
SqlDbType.NVarChar, 20).Value = strUsername

                If sdaLogin.SelectCommand.Connection.State =
ConnectionState.Open Then
                    blnResult = True
                Else
                    blnResult = False
                End If

            Catch ex As Exception
                strErrMsg = ex.Message
                blnResult = False
            End Try
            scnConnection.Close()

        End If
    End Function

    Public Function bindMemberLoginDatatable(ByRef strErrMsg As String,
ByRef dtLogin As DataTable, ByRef sdaLogin As SqlDataAdapter, ByRef
dvLogin As DataView,
   ByVal strUsername As String, ByVal strPassword As String) As Boolean

        Dim blnresult As Boolean
        Dim strHash As String
        Dim strTemp As String
        Dim intCat As Integer
        Try
            sdaLogin.Fill(dtLogin)
```

```
            If dtLogin.Rows.Count > 0 Then

                dvLogin = dtLogin.DefaultView
                strHash = dvLogin.Table.Rows(0).Item(2).ToString()
                strTemp =
FormsAuthentication.HashPasswordForStoringInConfigFile(strPassword,
"SHA1")

                If strHash = strTemp Then
                    blnresult = True
                End If
            Else
                blnresult = False
            End If
        Catch ex As Exception
            strErrMsg = ex.Message
            blnresult = False
        End Try

        Return blnresult
    End Function
```

Other class files in the system are:

- clsAdministrator
- clsCase
- clsDegree
- clsItemID

These class files contain functions directly related to the information that is needed by the system, rather being all in one single class file. In this way, the functions are in better order and arrangement and will not be difficult to be debugged or to trace errors.

6.3.2 PAS Screens

PAS contains 16 main screens that provide 4 major modules, Administration, Programmes, Recommendations and Users.

Administration modules consist of all the sub modules and functions to control and to administrate the operations of PAS. These include adding new degrees, editing degrees, view user records and also create and update PAS users and administrators. In the system, all these modules are available only in the Administration Screens which are only accessible by the system administrations and system owners only.

Programmes module consists of functions to display the list of degrees available to the users to choose and to learn more about each degree programme. This module is relatively small as it has only 2 screens to offer to the users.

Recommendation module consists of 4 major sub modules:

- Information Acquisition Sub Module
- Information Formatting and Classification Sub Module
- Similar Case Retrieval Sub Module
- Adaptation and Reuse Sub Module

The Information Acquisition Sub Module is in charge of acquiring user details and inputs to be processes in order to produce recommendations. While the Information Formatting and Classification Sub Module is in charge of segregating information retrieved either from the user or the database in order for easier processing and analysis.

The Similar Case Retrieval Sub Module's function is to retrieve cases similar to the problem information of the user. This retrieval process uses the nearest neighbour algorithm to retrieve cases from the database.

Finally, the Adaptation and Reuse Sub Module is the most important part in the Recommendation module where the cases and information gathered from the database will be adapted according to the needs of the user and is to be reuse to produce personalize recommendations to the user.

The Users module evolves around the functions to add and update member information and retrieval. This is to enable the users to join in as a member to use the facility provided and also update certain information if is found wrong.

6.3.2.1 Administration Module

- Initialize ()

```
Private Sub Initialize()
        Dim dtcase As New DataTable
        Dim blnResult As Boolean
        clsCase.bindAllCasesToday(strErrMsg, scnConnection, sdaCase)
        blnResult = clsCase.bindCaseDatatable(strErrMsg, dtcase,
sdaCase)
        If blnResult = True Then
            lblErrMsg.Visible = False
            wgRecords.DataSource = dtcase
            wgRecords.DataBind()
        Else
            If strErrMsg Is Nothing Then
                lblErrMsg.Text = "No Records"
            Else
                lblErrMsg.Text = strErrMsg
            End If
            lblErrMsg.Visible = True
        End If

 End Sub
```

- Save ()

```
Private Sub btnSave_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnSave.Click
        Dim strNew As String = ddlType.SelectedValue
        Dim strName As String = txtName.Text
        Dim strGender As String = ddlGender.SelectedValue.ToString
        Dim strRace As String = ddlRace.SelectedValue.ToString
        Dim strEmail As String = txtEmail.Text
        Dim strContact As String = txtContact.Text
        Dim strNRIC As String = txtNRIC.Text
        Dim strUsername As String = txUsername.Text
        Dim strPassword As String = txtPassword.Text
        Dim strID As String = lblID.Text
```

```
        Dim dtDOB As Date
        Dim blnResult As Boolean

        If wdcDOB.Value.ToString = "Null" Or wdcDOB.Value.ToString = ""
Then
            wdcDOB.Value = Date.Today
            dtDOB = wdcDOB.Value
        Else
            dtDOB = wdcDOB.Value
        End If

        If strNew = "Administrator" Then
            If Session("Action") = "Add" Then
                blnResult = clsAdministrator.newAdmin(strErrMsg,
scnConnection, sdaAdmin, strName, strEmail, strUsername, strPassword,
strContact)
                If blnResult = False Then
                    lblErrMsg.Visible = True
                    lblErrMsg.Text = strErrMsg
                Else
                    lblErrMsg.Visible = True
                    lblErrMsg.Text = "New user added!"
                    Initialize()
                    inactiveControls()

                    Session.Remove("Action")
                End If
            ElseIf Session("Action") = "Edit" Then
                blnResult = clsAdministrator.updateAdmin(strErrMsg,
scnConnection, sdaAdmin, strName, strEmail, strContact, strID)
                If blnResult = False Then
                    lblErrMsg.Visible = True
                    lblErrMsg.Text = strErrMsg
                Else
                    lblErrMsg.Visible = True
                    lblErrMsg.Text = "Update Succesful!"
                    Initialize()
                    inactiveControls()

                    Session.Remove("Action")
                End If
            End If

        ElseIf strNew = "Student" Then
            If Session("Action") = "Add" Then
                blnResult = clsMember.newMember(strErrMsg,
scnConnection, sdaMember, strName, strEmail, strUsername, strPassword,
strNRIC, dtDOB, strGender, strRace)
                If blnResult = False Then
                    lblErrMsg.Visible = True
                    lblErrMsg.Text = strErrMsg
                Else
                    lblErrMsg.Visible = True
                    lblErrMsg.Text = "New user added!"
                    Initialize()
                    inactiveControls()
```

```
                    Session.Remove("Action")
                End If
            ElseIf Session("Action") = "Edit" Then
                blnResult = clsMember.updateMember(strErrMsg,
scnConnection, sdaMember, strName, strEmail, dtDOB, strGender, strRace,
strID, strNRIC)
                If blnResult = False Then
                    lblErrMsg.Visible = True
                    lblErrMsg.Text = strErrMsg
                Else
                    lblErrMsg.Visible = True
                    lblErrMsg.Text = "Update Succesful!"
                    Initialize()
                    inactiveControls()

                    Session.Remove("Action")
                End If
            End If

        Else
            lblErrMsg.Visible = True
            lblErrMsg.Text = "Error! Invalid Registration!"
        End If


    End Sub
```

- ActiveRowChange ()

```
Private Sub wgUsers_ActiveRowChange(ByVal sender As Object, ByVal e As
Infragistics.WebUI.UltraWebGrid.RowEventArgs) Handles
wgUsers.ActiveRowChange
        If ddlSelect.SelectedValue = "spAllAdmin" Then

            lblID.Text = e.Row.Cells(0).ToString
            txtName.Text = e.Row.Cells(3).ToString
            txtContact.Text = e.Row.Cells(5).ToString
            txtEmail.Text = e.Row.Cells(4).ToString
            txtNRIC.Text = ""
            txtPassword.Text = e.Row.Cells(2).ToString
            txUsername.Text = e.Row.Cells(1).ToString
            ddlRace.SelectedValue = "Malay"
            ddlGender.SelectedValue = "Male"
            ddlType.SelectedValue = "Administrator"
            wdcDOB.Value = ""
            btnEdit.Enabled = True

        ElseIf ddlSelect.SelectedValue = "spAllStudent" Then

            lblID.Text = e.Row.Cells(0).ToString
            txtName.Text = e.Row.Cells(3).ToString
            txtContact.Text = ""
            txtEmail.Text = e.Row.Cells(6).ToString
            txtNRIC.Text = e.Row.Cells(5).ToString
            txtPassword.Text = e.Row.Cells(2).ToString
            txUsername.Text = e.Row.Cells(1).ToString
            ddlRace.SelectedValue = e.Row.Cells(7).ToString
            ddlGender.SelectedValue = e.Row.Cells(8).ToString
            ddlType.SelectedValue = "Student"
            wdcDOB.Value = e.Row.Cells(4).ToString
            btnEdit.Enabled = True

        ElseIf ddlSelect.SelectedValue = "spAllUser" Then

            lblID.Text = e.Row.Cells(0).ToString
            txtName.Text = e.Row.Cells(3).ToString
            txtContact.Text = ""
            txtEmail.Text = e.Row.Cells(5).ToString
            txtNRIC.Text = ""
            txtPassword.Text = e.Row.Cells(2).ToString
            txUsername.Text = e.Row.Cells(1).ToString
            ddlRace.SelectedValue = "Malay"
            ddlGender.SelectedValue = "Male"
            ddlType.SelectedValue = "Lecturer/Councellor"
            wdcDOB.Value = ""
            btnEdit.Enabled = True

        End If
End Sub
```

6.3.2.2 Programme Module

- bindSearch()

```
Private Sub bindSearch(ByVal strSearch As String)

        Dim dtSearch As New DataTable
        Dim dvSearch As New DataView
        Dim blnResult As Boolean

        With clsDegree
            blnResult = .bindDegree(strErrMsg, scnConnection,
sdaDegree)
            If blnResult = False Then
            Else
                blnResult = .bindDegreeTable(strErrMsg, dtSearch,
sdaDegree)
            End If

        End With
        If blnResult = False Then
            lblErrMsg.Visible = True
            lblErrMsg.Text = "No Records Found!"
        Else
            lblErrMsg.Visible = False

        End If

        dvSearch = dtSearch.DefaultView

        dvSearch.RowFilter = "degreeName like '%" & strSearch & "%' or
fieldName like '%" & strSearch & "%' "

        wgDegree.DataSource = dvSearch
        wgDegree.DataBind()

    End Sub
```

- convertScore()

```
Private Function convertScore(ByVal strScore As String) As Double
        Dim strFront As String
        Dim strBack As String
        Dim dblValue As Double
        Dim intCount As Integer

        intCount = strScore.Length()

        If intCount = 2 Then
            strFront = strScore.Substring(0, 2)
            dblValue = CType(strFront, Double)

        ElseIf intCount = 3 Then
            strFront = strScore.Substring(0, 2)
            dblValue = CType(strFront, Double)

            strBack = strScore.Substring(3, 1)
            dblValue = dblValue + (CType(strBack, Double) / 100)
```

```
        ElseIf intCount >= 4 Then
            strFront = strScore.Substring(0, 2)
            dblValue = CType(strFront, Double)

            strBack = strScore.Substring(3, 2)
            dblValue = dblValue + (CType(strBack, Double) / 100)

        End If

        Return dblValue


    End Function
```

6.3.2.3 Recommendation Module

- btnProceed_Click()

```
Private Sub btnProceed_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnProceed.Click

        Dim dblSub1 As Double = CType(ddlSub1.SelectedValue.ToString,
Double)
        Dim dblSub2 As Double = CType(ddlSub2.SelectedValue.ToString,
Double)
        Dim dblSub3 As Double = CType(ddlSub3.SelectedValue.ToString,
Double)
        Dim dblSub4 As Double = CType(ddlSub4.SelectedValue.ToString,
Double)
        Dim dblTotal As Double = (dblSub1 + dblSub2 + dblSub3 +
dblSub4) * 1.125

        Session("Total") = dblTotal
        Session("Stream") = ddlStream.SelectedValue
        If dblTotal < 70.0 And dblTotal > 51 Then
            lblErrMsg.Visible = True
            lblErrMsg.Text = "Note : Based on your results, the system
may be unable to recommend a suitable degree programme for you."
            btnContinue.Visible = True
            btnProceed.Visible = False
        ElseIf dblTotal <= 50 Then
            lblErrMsg.Visible = True
            lblErrMsg.Text = "Sorry!System is unable to recommend a
suitable degree programme for you."
            btnProceed.Visible = False
        Else
            Response.Redirect(Request.ApplicationPath &
"/Screens/Recomm/QnA.aspx")
        End If

End Sub
```

- selfRecommendation()

```
Private Sub selfRecommendation(ByVal dblTotal As Double, ByVal strQ1 As
String, ByVal strQ2 As String, ByVal strQ3 As String, ByVal strQ4 As
String)
        Dim blnresult As Boolean
        Dim dtRecommend As New DataTable
        Dim dvCases As New DataView

        With clsCase
            .bindPossibleCase(strErrMsg, scnConnection, sdaCase,
dblTotal)
            blnresult = .bindCaseDatatable(strErrMsg, dtRecommend,
sdaCase)
            If blnresult = False Then
                lblErrMsg.Visible = True
                lblErrMsg.Text = "Error! Database Problem! Please
contact your Administrator!"
            Else

                dvCases = dtRecommend.DefaultView
                dvCases.RowFilter = "(req1 = '" & strQ1 & "' and req2=
'" & strQ2 & "' and req3= '" & strQ3 & "' and req4='" & strQ4 & "' and
matchValue >= 0.5 and matchValue <=1.00  )"

                If dvCases.Count > 0 Then
                Else
                    dvCases.RowFilter = "(req1 = '" & strQ1 & "' and
req2= '" & strQ2 & "' and req3= '" & strQ3 & "' or req4='" & strQ4 & "'
and matchValue >= 0.5 and matchValue <=1.00 )" _
                    & "or  (req1 = '" & strQ1 & "' and req2= '" &
strQ2 & "' or req3= '" & strQ3 & "' and  req4='" & strQ4 & "' and
matchValue >= 0.5 and matchValue <=1.00 )" _
                    & "or (req1 = '" & strQ1 & "' or req2= '" & strQ2
& "' and req3= '" & strQ3 & "' and req4='" & strQ4 & "' and matchValue
>= 0.5 and matchValue <=1.00 )  "
                End If
                Session("RecommendSys") = dvCases.Table.DefaultView

                Response.Redirect(Request.ApplicationPath &
"/Screens/Recomm/RecommendationResult.aspx")
            End If
        End With
    End Sub
```

- Initialize()

```
Private Sub Initialize()
        Dim dtMember As New DataTable
        Dim dtRecomm As New DataTable
        Dim blnResult As Boolean
        Dim strUser As String
        Dim strusername As String = Session("User")

        clsMember.bindMemberInfoAdapter(strErrMsg, scnConnection,
sdaMember, strusername)
        clsMember.bindMemberInfoDatatable(strErrMsg, dtMember,
sdaMember)
```

```
        lblUsername.Text = dtMember.Rows(0).Item(3).ToString
        strUser = dtMember.Rows(0).Item("studentID").ToString

        clsCase.bindUserCase(strErrMsg, scnConnection, sdaCase,
strUser)
        clsCase.bindCaseDatatable(strErrMsg, dtRecomm, sdaCase)

        If Session("FinalRecomm") Is Nothing Then
            If dtRecomm.Rows.Count > 0 Then
                bindPage(dtRecomm)
            Else
                lblErrMsg.Visible = True
                lblErrMsg.Text = "Error!No existing records found!"
            End If

        Else
            lblErrMsg.Visible = False
            dtRecomm = Session("FinalRecomm")

            bindPage(dtRecomm)

        End If

        If Session("Total") Is Nothing Then
            If dtRecomm.Rows.Count > 0 Then
                lblScore.Text =
dtRecomm.Rows(0).Item("caseTotal").ToString
            Else
                lblScore.Text = "No Score Available"
            End If

        Else
            lblScore.Text = Session("Total")
        End If
    End Sub
```

6.3.2.4 Users Module

- checkSession()

```
Private Sub checkSession()
        If Session("Update") = "Updated" Then
            lblErrMsg.Visible = True
            lblErrMsg.Text = "Your changes has been updated
successfully!"
            Session("Update") = ""
        ElseIf Session("Update") = "Failed" Then
            lblErrMsg.Visible = True
            lblErrMsg.Text = "Error! Your changes has not been updated
yet!"
            Session("Update") = ""
        ElseIf Session("Update") = "" Then
            lblErrMsg.Visible = False
        End If
    End Sub
```

- btnSave_Click()

```
Private Sub btnSave_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnSave.Click
        Dim strName As String = txtName.Text
        Dim strEmail As String = txtEmail.Text
        Dim strNRIC As String = txtNRIC.Text
        Dim dtDOB As Date
        Dim strGender As String = rblSex.SelectedValue.ToString
        Dim strRace As String = rblRace.SelectedValue.ToString
        Dim strID As String = lblID.Text
        Dim blnResult As Boolean


        If wdcDOB.Value.ToString = "Null" Or wdcDOB.Value.ToString = ""
Then
            wdcDOB.Value = Date.Today
            dtDOB = wdcDOB.Value
        Else
            dtDOB = wdcDOB.Value
        End If

        blnResult = clsMember.updateMember(strErrMsg, scnConnection,
sdaMember, strName, strEmail, dtDOB, strGender, strRace, strID,
strNRIC)
        If blnResult = False Then
            Session("Update") = "Failed"
        ElseIf blnResult = True Then
            Session("Update") = "Updated"
        End If

        Response.Redirect(Request.ApplicationPath &
"/Screens/User/Main.aspx")

    End Sub
```

6.3.3 PAS Screenshots

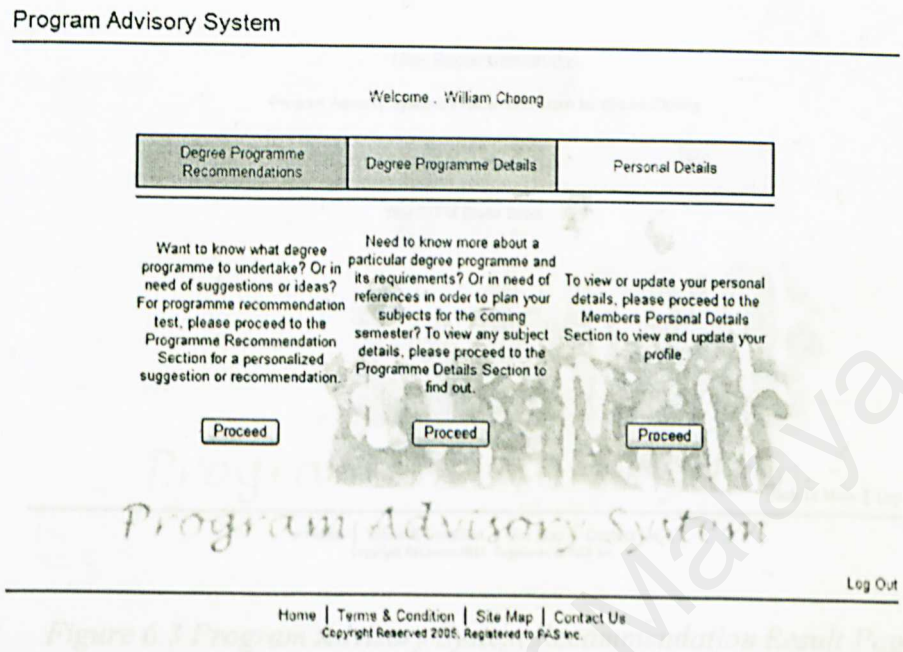


Figure 6.1 Program Advisory System Member's Main Page

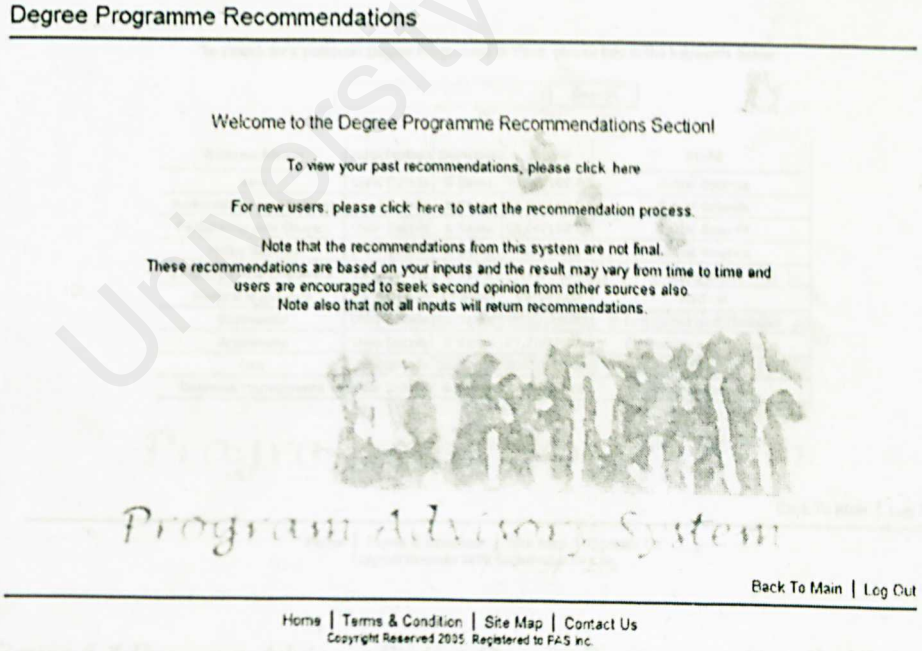


Figure 6.2 Program Advisory System Recommendation Page

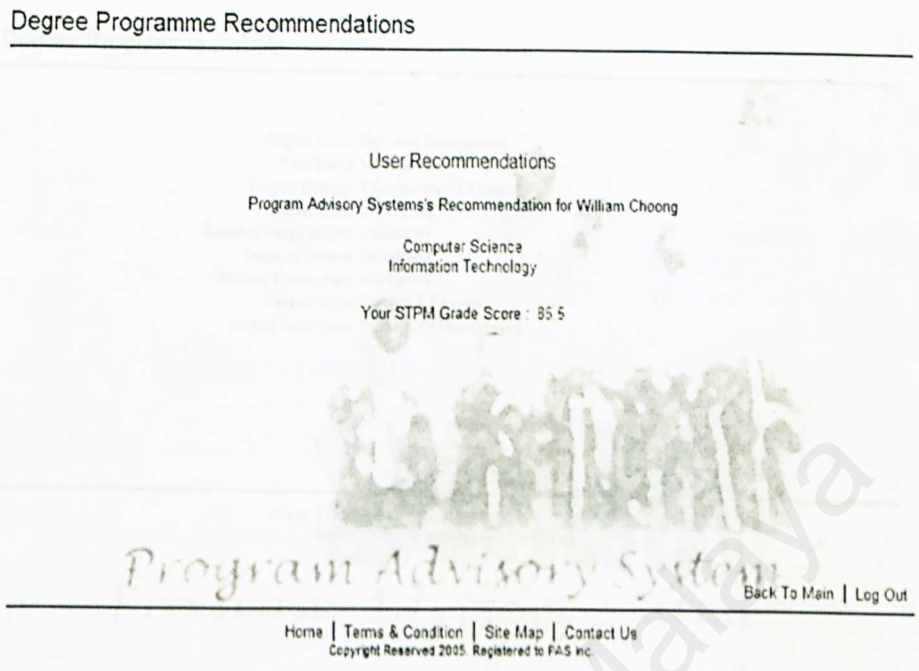


Figure 6.3 Program Advisory System Recommendation Result Page

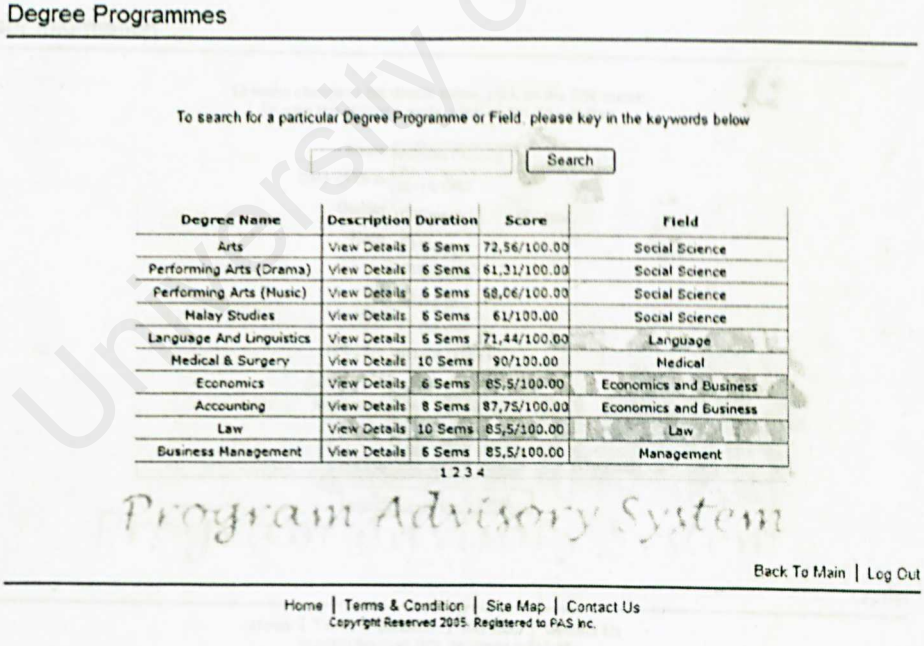


Figure 6.4 Program Advisory System Degree Programme Details Main Page

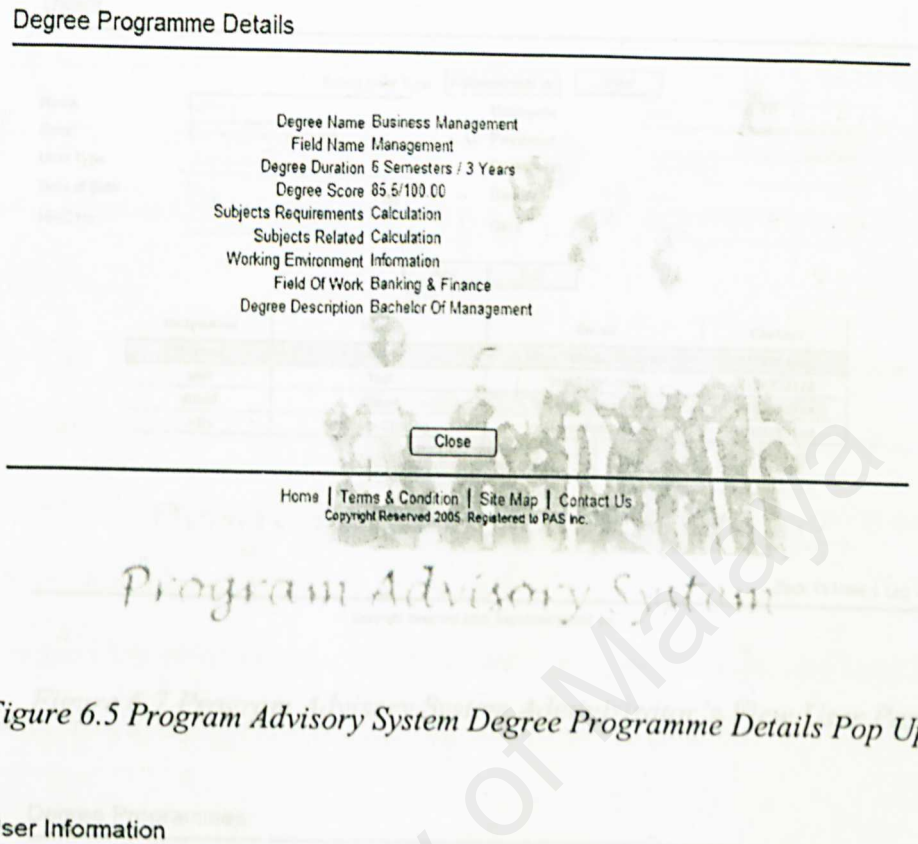


Figure 6.5 Program Advisory System Degree Programme Details Pop Up

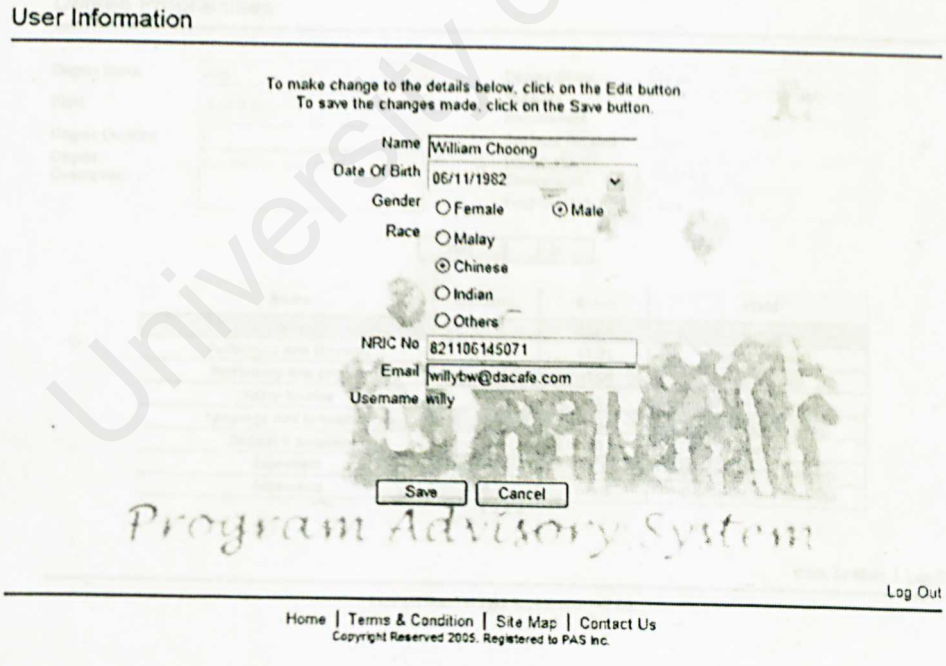


Figure 6.6 Program Advisory System User Information Edit Page

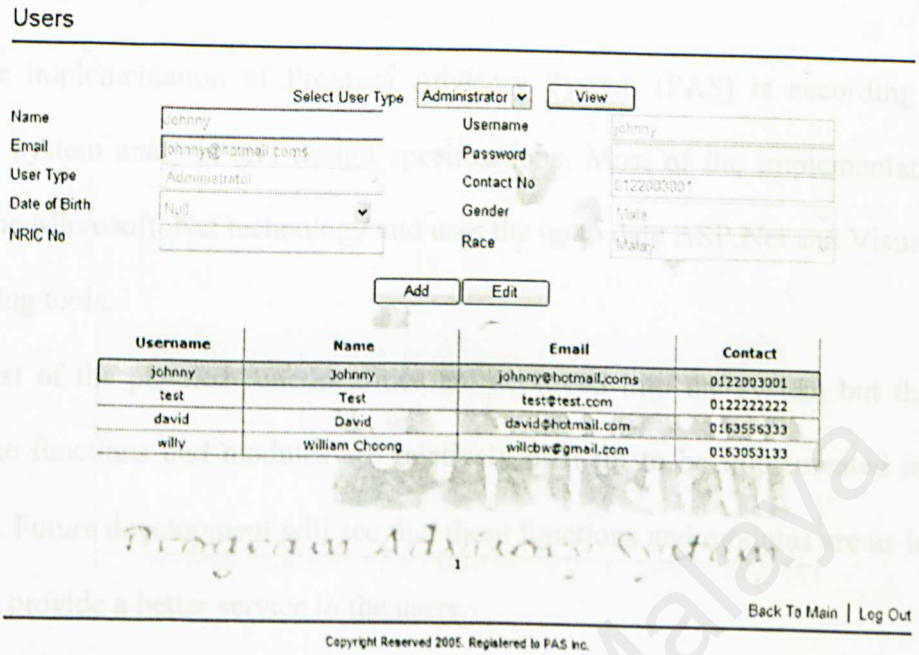


Figure 6.7 Program Advisory System Administrator's View User Page

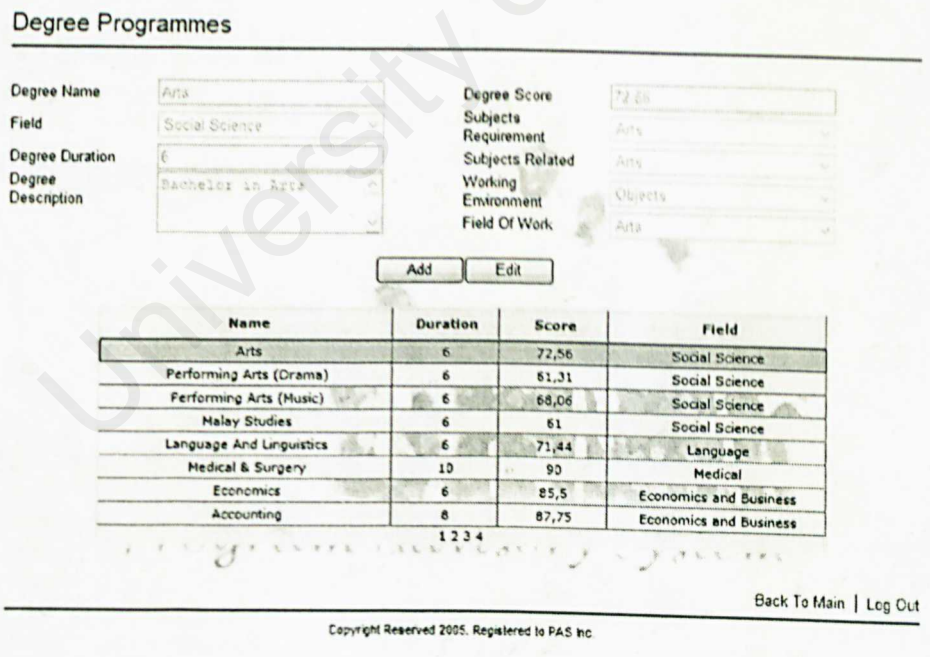


Figure 6.8 Program Advisory System Administrator's Degree Programmes Main Page

6.4 Chapter Summary

The implementation of Program Advisory System (PAS) is according to the predefined system analysis and design specifications. Most of the implementation are based on the Microsoft .Net technology and uses the up to date ASP.Net and Visual Basic programming tools.

Most of the planned functionalities are deployed into the system but there are some of the functions and modules are relatively difficult to be implemented in a full scale basis. Future development will see that these functions and modules are as in depth it can be to provide a better service to the users.

The next following chapter will talk about the system testing and how it can test the successfulness of the system implementation and development.

Chapter 7: System Testing

Chapter 7: System Testing

7.1 Introduction

System testing is one of the most important stages in the software development life cycle as it determines the functionality, quality and performance of the software developed meets the requirements from the system analysis and design stages.

A system should be tested throughout its development, not only during after implementing and integrating the whole system. This is due to the fact that by the end of the implementation of the system, whichever system defect and errors that needed to be repaired and fix will take a longer time and waste important resources, as it would affect the system in the whole. But if testing was to be performed from time to time during the implementation stages, errors can be fix much easier and also before system integration is done where it will not affect the whole system's processes.

For the Program Advisory System, I have implemented the testing phases, as shown below:

- Unit Testing
- Module Testing
- Integration Testing
- System Testing

7.2 Unit Testing

Unit testing is done by testing the smallest components of the each and every one of the modules in PAS. Testing is done based on 2 major unit testing techniques:

- Black Box Testing
- White Box Testing

Black box testing is based solely on the functionality of the codes in the system. The tests are designed and planned to check for the output of the functions, whether they comply with the expected results based on the inputs given.

This testing technique however requires millions of test cases to fully and exhaustively test a system. This is due to the fact that each and every test case must be able to detect undetected fault. But this type of testing is free from any logical standpoint and constraints as the internal structure of the system can be ignored.

White box testing is about examining the codes rather than the specifications of the system. The testing is done by analyzing the codes and then ensuring all of the tests cover all statements, branch and path available. The execution of each of the statements for instance is vital to ensure the function of the system is done correctly.

For PAS, both the black box testing and the white box testing is done hand in hand through out the implementation of the system. Each unit testing consist of both black box and white box testing to test different aspects of the system, where black box testing can be used to test for equivalence and boundary value testing while white box testing can be used to test for statement and path coverage test.

7.3 Integration Testing

Integration testing is another testing phase executed after the integration of components that would make up a certain module. After each of the smaller components undergo exhaustive and pre planned unit testing, they will be integrated into modules and during after the integrations the integration testing is performed.

This phase of testing will test the overall flow of the units combined together. This is due to the fact that when the smaller units are combined together, their interaction would cause some unexpected errors to occur even though during unit testing the smaller components worked well in the test cases. In this way, integration errors can be detected earlier and prevention measures can be done before it is too late.

However, integration testing done will be strongly influenced by the type of system that is being implemented. This is due to the fact that systems with graphical user interfaces are tested slightly different then those without graphical user interfaces. So, different CASE tools are needed to monitor the integration testing procedures.

For PAS, integration testing will be done in the Top Down approach where the test cases will extensively test the modules from the top of the tier until the bottom most tiers.

7.4 System Testing

The final testing phase is the system testing phase, where the system as a whole will be tested. The system will be tested based on the user requirements, whether the outcome of the system meets the user's needs. The system must be tested to ensure that it performs well and within the expectations of the users.

The robustness and the security aspects of the system are also important and should be tested in this phase. The stress test and load test should also be done in this phase to show how the system will react to the traffic and load as if it is being deployed.

7.5 List of Tests Performed

During the system testing of PAS, I have conducted a series of individual and combined tests to validate the system implemented according to the system analysis and design phases.

The list is as follows:

- Input Validation Test
- Input Boundary Test
- Required Information Test
- Database Connectivity Test
- Database Retrieval And Validation Test
- Error Control Test
- Security Test

7.6 Chapter Summary

System Testing is one of the final phases in the software development process, but it implementation coincide with the implementation and the development of the system itself, and not the after the system has been developed and integrated.

The PAS system has gone through many repetitious tests again and again to enable the developers and the system testers to check the system's functionality, robustness, the overall response time and also integrity other than what is required in the user requirements documentation. Initially, the system produced a number of complications and error but through the extensive testing plans and cases, these complications and errors were detected and rectified in the short period of time.

PAS has also shown a great percentage of reliability in the sense of the performance during critical errors, like for instance the loss of database connection or perhaps unauthorized access to the system.

In the next and final chapter, the report will discuss about the system's strengths and weakness and also an overall conclusion of the whole system itself.

Chapter 8: System Evaluation and Conclusion

8.1 System Evaluation
8.2 Conclusion

Chapter 8: System Evaluation and Conclusion

8.1 System Evaluation

System Evaluation is the process where a completed system is evaluated for certain key criteria, for example whether the system has reached its intended goal or target, whether the system has performed its task as required or stated in the user requirements, its limitations and it capabilities.

The evaluation for PAS will be divided into 4 parts, mainly:

- Overall Result Overview
- Problems Encounter and Solutions
- Strengths and Weaknesses of the System
- Future Enhancements of the System

8.1.1 Overall Result Overview

The overall result of the system analysis, design, implementation and testing has been encouraging. Although the whole process is being carried out on a short period of time, but the overall system being developed has reached its main goal that is to be able to recommend the users degree programmes that is suited to their personality or interest, on top of their overall results.

The system has performed well under critical situation and has shown security and integrity in handling important and private information.

8.1.2 Problems Encounter and Solutions

There are also some major problems encountered during the course of developing the system. They are:

- **Defining the scope of the system**

  The Program Advisory System covers a very wide scope, in the sense of the types of education and qualifications that makes up the whole education standards in Malaysia. On top of that, there are many types of degree programmes that are similar in name, but different in the requirements, duration and also scope.

  But for PAS, the main scope of the whole system evolves around STPM school leavers who would like to enter local public universities.

- **Implementing the Case based Reasoning algorithm**

  The implementation of the case based reasoning algorithm was slightly difficult, based on the type of development tools used. Moreover, the criteria taken into consideration for the calculation of the case based reasoning algorithm is also very important to produce a good and reliable value.

- **Repetitious system testing**

  Extensive system testing was done towards the end of the development stages, as most of the integration works are done towards the end of the development and implementation of the system. Other that the usual unit testing, integration testing and the overall system testing must be done intensively and in detail to track down system errors and critical errors that would eventually cause the whole system cease its operations abruptly.

The overall testing phase was very time consuming but it was completed successfully towards the end of the allocated time.

8.1.3 Strengths and Weakness of the System

Program Advisory System has its own list of capabilities and weaknesses, due to time constraints and limited domain knowledge.

Strengths

- **System Transparency**

  The Program Advisory System is transparent in its operations, meaning that the user need not to know how the whole system functions, what type of services it needs to be able to run the system and how the calculations are made. The user only needs to interact with the graphical user interface of the system and need not to toggle or execute certain codes to be able to use the system.

- **Error Handling**

  Program Advisory System has an extensive error handling functions where any errors made will be display to the user. This will curb from any abrupt stoppages when the user is using the system itself. Each different errors will produce different personalize error messages.

- **Easy to use**

  The simple and relatively easy to use graphical user interface is also one of the strengths of the Program Advisory System. The design and the colour contrast of the system has been carefully selected to make the interface more appealing and in the same time less complicated to be used by non computer savvy users.

- **Security**

  The system also has implemented security measure in order to promote system integrity and privacy. The system uses a session function to track the user's interaction with the system, and when the user is idle for more than 10 minutes, the system will by itself terminate the user's session in order to keep the user's information safe and intact. The system also has security measure under place for unauthorized access to the system by having log in sessions, redirection of users to the main guest area when the user session has expired and also user input checking functions.

Weaknesses

- **Limited range of degree programmes**

  Currently, the list of degree programmes stored in the Program Advisory System's repository is limited, due to the fact that the degree programmes selected are those which are more popular and in demand. There are also overlapping degree programmes that are left out, as it actually describes the same type of degree programme, but only the difference in requirements and duration.

- **Limited range of user evaluation questions**

  The system has only a limited range of evaluation questions for the users to complete and this would slightly affect the course of the recommendation process. However, the current system now takes only the most important questions that would directly lead towards the main intention of streaming out the interest of the user.

8.1.4 Future Enhancements of the System

In the future, more feature are planned in order to make the system more effective and provide more depth in its services.

Due to the time constraints and the man power, these enhancements are supposed to be integrated earlier in the system. But in the coming months, these enhancements will be part and parcel of the Program Advisory System

- **Improved Graphical User Interface**

  One of the pre planned enhancement of the system is to improve the graphical user interface to suit multiple web browsers and to promote a more personalize feel to each and every single user.

  The users will be able to choose the basic appearance of the system and also be more comfortable with the controls in the system.

- **Multiple user level recommendations**

  The future enhancement of the Program Advisory System will see the increase of the target users of the system, where in the future the system will be able to cater for other levels of education of the user, for instance SPM school leavers, matriculation level students and also A Levels level students.

- **Multiple language support**

  Program Advisory System should, in the future cater for all the major language groups in Malaysia, mainly for the Malay community, Chinese community and the Indian community, by offering the system in 4 major languages; English, Malay Language, Mandarin and also Tamil.

- **More intensive recommendations**

  The Program Advisory System's recommendation process will be enhanced in the future to analyze more inputs from the user, in order to produce a better and more accurate recommendation to the user. The case based reasoning algorithm will be refined and will take in more parameter into consideration.

8.2 Conclusion

Program Advisory System is a case based advisory programme that recommends its user degree programmes for tertiary education. Its web based application makes it accessible from anywhere in the world, with the help of the internet technology.

The implementation of the Program Advisory System is based on the Microsoft .Net technology, mainly the ASP.Net front end development tool coupled with the Visual Basic back end developing tool together with other important Microsoft application and services. It also incorporates JavaScript programming language into the development and control over the system.

The main idea of the Program Advisory System is recommend and at the same time appear as a basic guideline and idea of what type of degree programme should the user undertake, given the set of inputs from the user. It incorporated cased based reasoning algorithm, a popular technique used in artificial intelligence field where existing and similar case solutions are adapted and suited to the problem case. These complicated algorithm are made simple by this system, where the implementation of the system is transparent, not needing the user to first understand how everything works.

The implementation life cycle of the Program Advisory System is the rapid prototyping life cycle model, where the prototype is developed to capture the user requirements real time and the rest of the implementation of the actual working system will be based on the general and specific ideas gathered from the initial prototype. This is a better way to extract the user's need directly as the more conventional way will be hazardous where the system owner will only be able to catch a glimpse of the system after which the system has been completed, where as for the rapid prototyping technique,

the system owners can define what they want in the system through the prototype developed.

During the course of the implementation of the Program Advisory System, much is gained through the hands on experience and the knowledge acquired along the way. Through this system development also, the knowledge gained from the past courses taken, namely Expert System course, System Analysis and Design course, Software Engineering course and many more.

Programming techniques and trend are also gained from the implementation of this system, and direct approach on some of the up to date development tools, like for instance Visual Studio .Net, Microsoft SQL Server 2000 and Adobe Photoshop CS proved to be beneficial in the future.

Proper system development planning is also one of the most important experiences gained from the development of the system, where meeting datelines and deliverables are very important to ensure the success of the system.

As a conclusion, the Program Advisory System is successfully developed and is an excellent system to be implemented as it proves it worth in giving its users a guideline to a successful career for them. Implementation of the system has been a fruitful experience and further enhancement of the system later on will make the Program Advisory System more a complete and round advisory system.

References

References

Luger, G.F. (2002). *Artificial Intelligence: Structures and Strategies for Complex Problem Solving.* 4th Ed. Pearson Education Limited.

Connolly, T., Begg, C. (2002). *Database Systems.* 3rd Ed. Pearson Education Limited.

Durkin, J. (1994). *Expert Systems: Design and Development.* Macmillan Publishing Company.

Schach, S.R. (2005). *Object – Oriented and Classical Software Engineering.* 6th Ed. McGraw Hill Higher Education.

Binh, V. N. (unknown). *AAA: Another Academic Advisor using Case – Based Reasoning.* PhD Thesis. Ohio University.

Moore, J.L., Means, T., Kim, B. (unknown). *Applying Case – Based Reasoning Within A Technology Integration Learning Environment.* Technical Paper. University of Missouri – Columbia.

Shephard, M. (unknown). *Case – Based Reasoning and Software Engineering*. Technical Report. Bournemouth University.

unknown,(2004).*A – Z of Courses Explained*.3rd ed. GTI Specialist Publishers.

Internet References:

- (URL – http://accounting.rutgers.edu/raw/aies/www.bus.orst.edu/faculty/ brownc/aies/newslet/fall95/casebase.htm), 20/7/2004

- (URL – http://www.geocities.com/andrewbroad/cs/cbr/cs3411.html), 20/7/2004

- (URL – http://danpatch.ecn.purdue.edu/~engelb/abe565/cbr.html), 20/7/2004

- (URL – http://www.i-content.org/GDA/CBR.html), 20/7/2004

- (URL – http://www.woodger.ca/archmult.htm), 20/9/2004

Appendix A : User Manual For Program Advisory System

User Manual for Program Advisory System

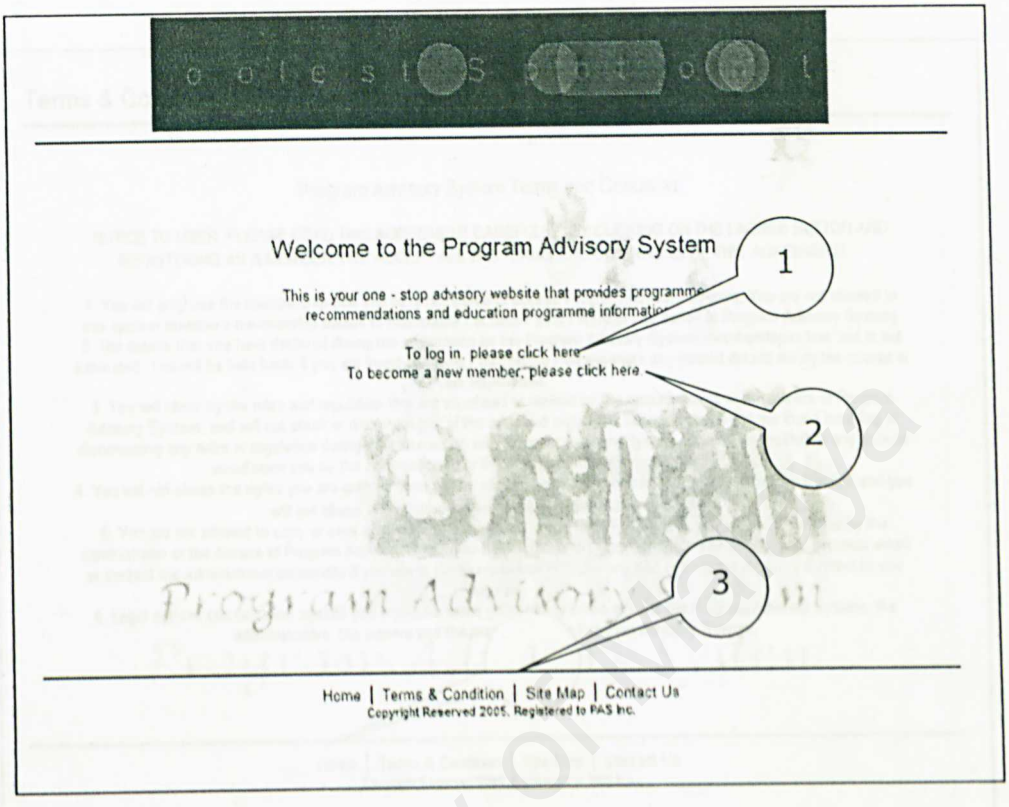
Program Advisory User Manual is designed to assist users on the operational aspect of the Program Advisory System. In this User Manual, all of the system's screens will be covered and step by step guide will be available for the users to review whenever a problem arises.

This manual also help the users to understand the system better and also helps to troubleshoot some of the common mistakes made by novice users.

This manual will be separated into 2 sections; one section is for the Program Advisory System Members while next section will be for the Program Advisory System Administrators.

Section A: Program Advisory System Member Screens

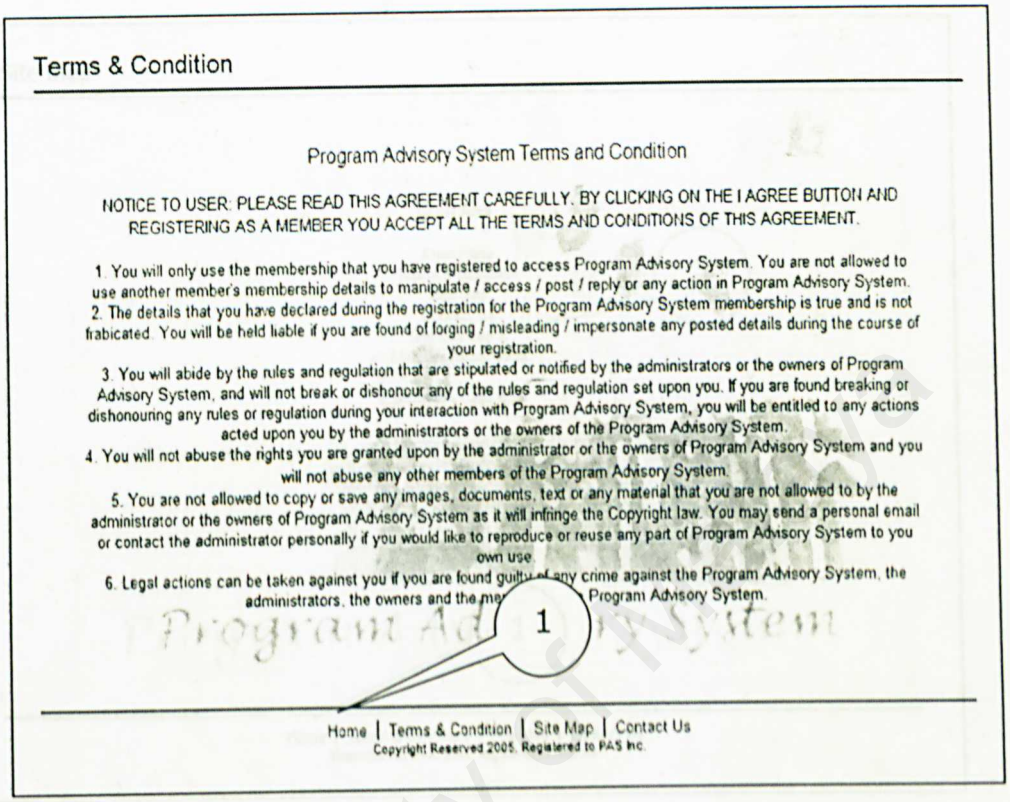
Main Screen



Instructions:

1. To access the log in page, click on link here.
2. To access the member registration screen, click on link here.
3. To access the Terms & Condition, Site Map and Contact Us screens, click on either on of the link here.

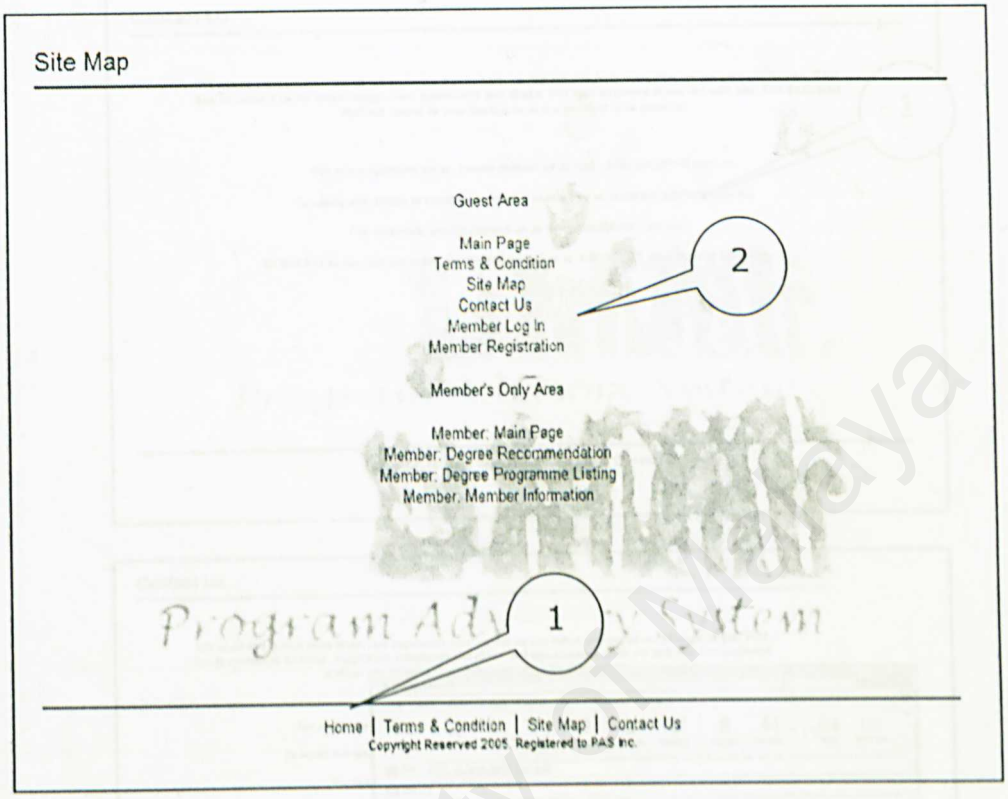
Terms & Condition Screen



Instructions:

1. To access the home page again, click on the link 'Home'.

Site Map Screen

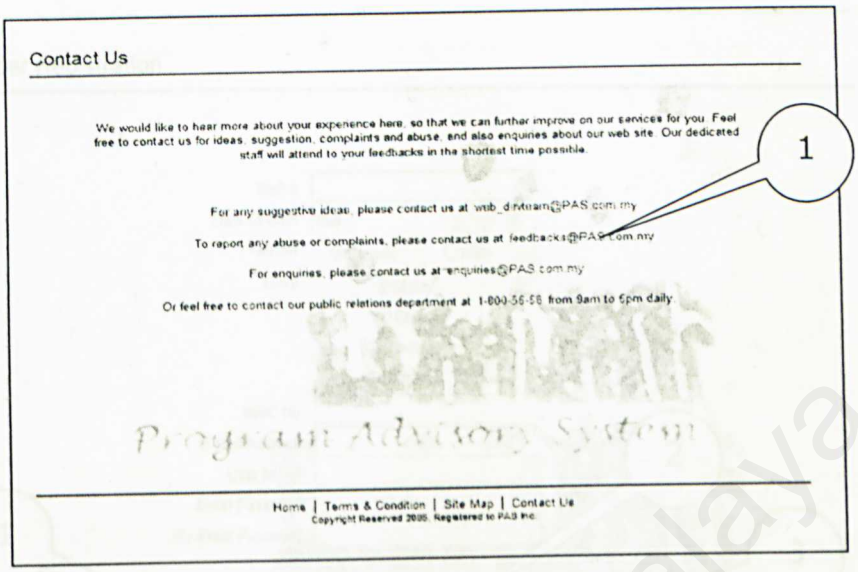


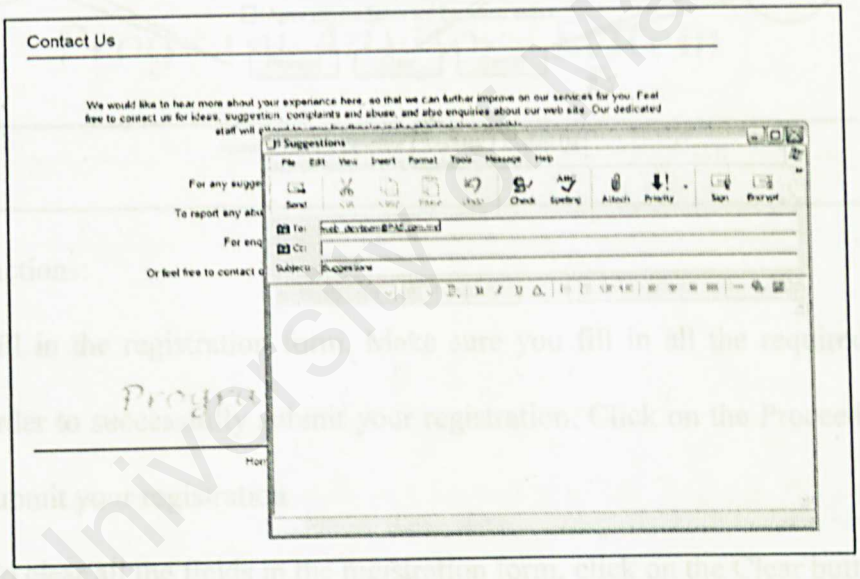
Instructions:

1. To access the home page again, click on the link 'Home'.
2. To access the links in the Guest Area or in the Member's Only Area, click on either one of the links provided.

Note: To access the Member's Area, you have to log into the Program Advisory System first. If not, it will redirect you straight to the Member's Log In screen.

Contact Us Screen

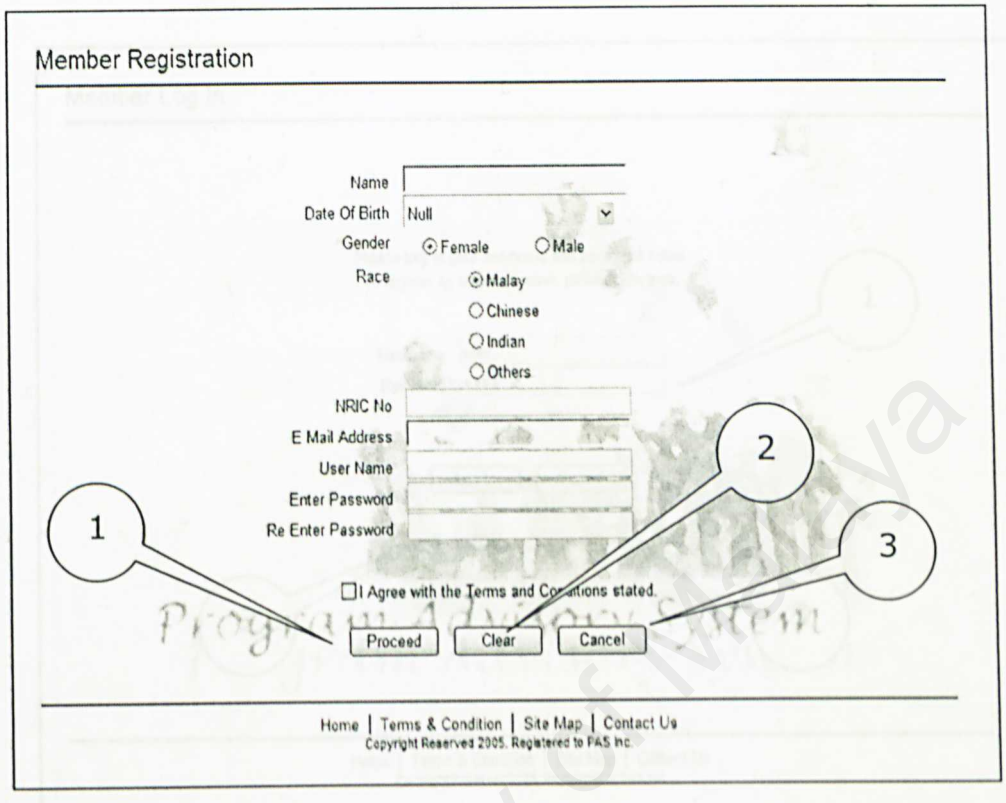




Instructions:

1. To send an email to the respective recipients, click on either one of the mailing links. It will trigger the Outlook Express to start up and you are ready to write your email.

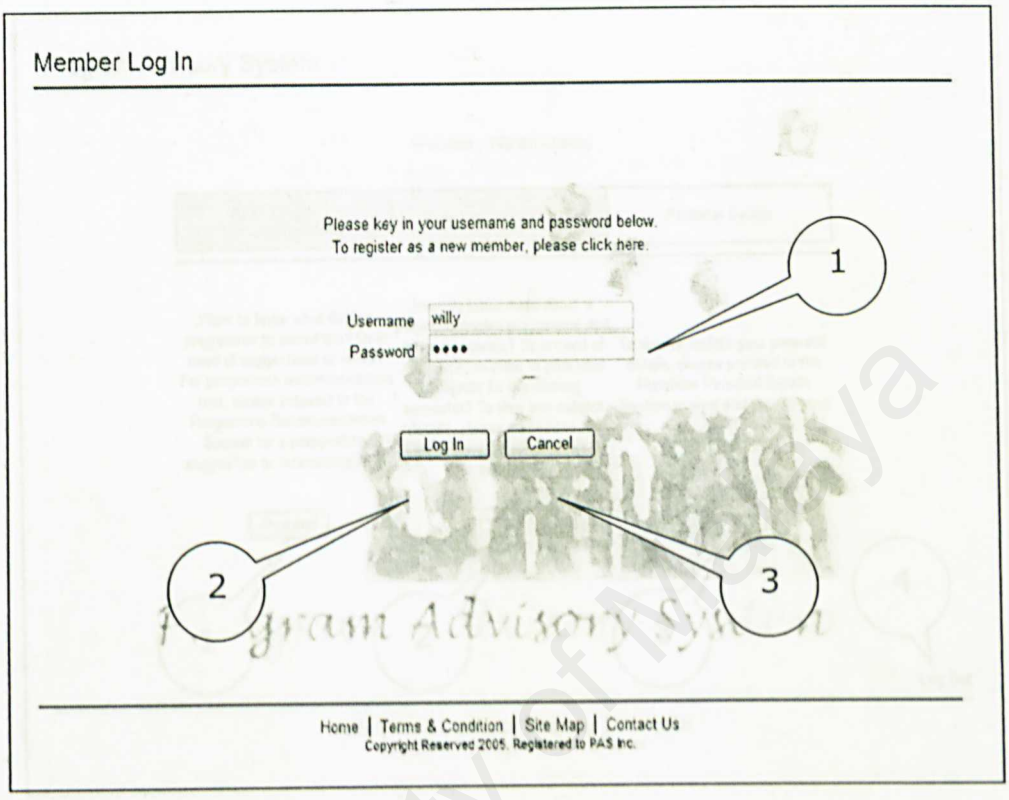
Terms & Condition Screen



Instructions:

1. Fill in the registration form. Make sure you fill in all the required fields in order to successfully submit your registration. Click on the Proceed button to submit your registration
2. To clear all the fields in the registration form, click on the Clear button.
3. To cancel your registration, click on the Cancel button to redirect you back to the Program Advisory System Main screen.

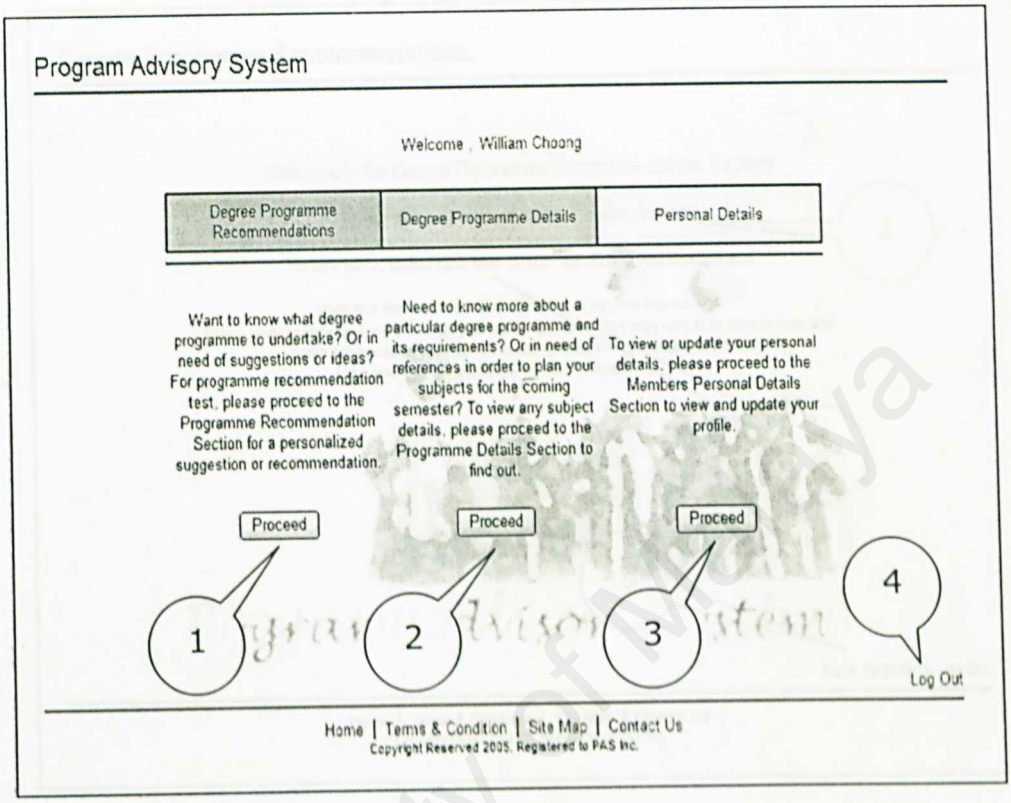
Member's Log In Screen



Instructions:

1. Fill in the username field and the password field.
2. To submit your username and password to log into the Program Advisory System, click on the Log In button.
3. To cancel your registration, click on the Cancel button to redirect you back to the Program Advisory System Main screen.

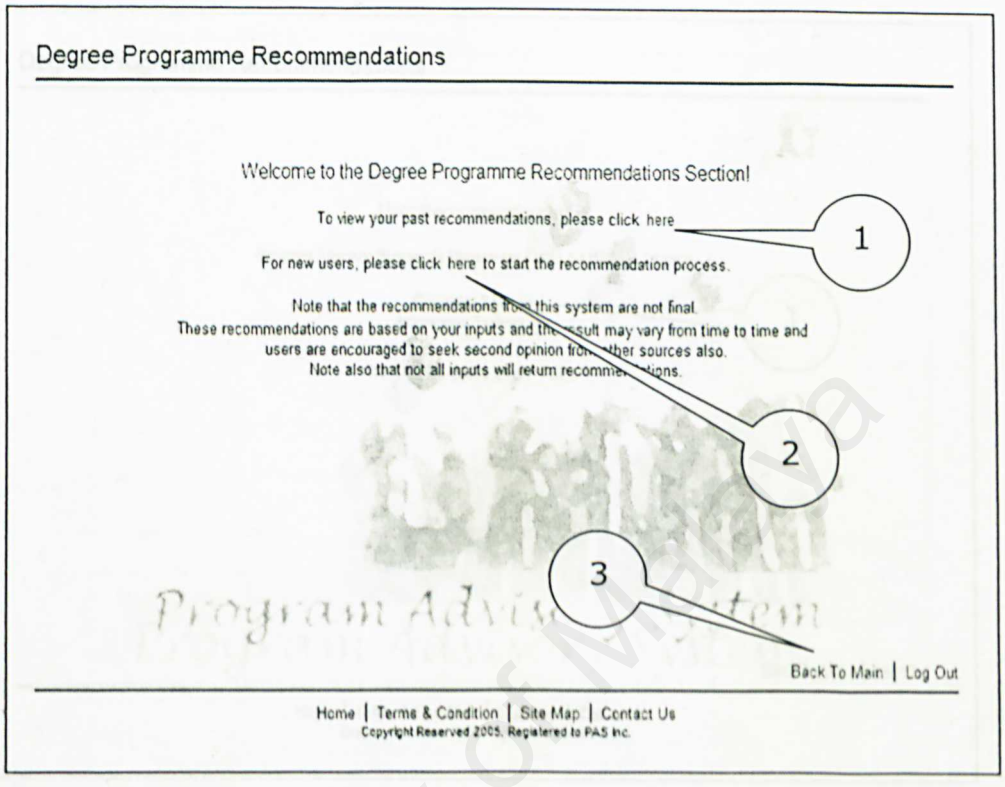
Member's Main Screen



Instructions:

1. To go into the Degree Programme Recommendation section, click on the first Proceed button.
2. To go into the Degree Programme Details section, click on the second Proceed button.
3. To go into the Personal Details section, click on the third Proceed button.
4. To log out of the Program Advisory System, click on the Log Out link.

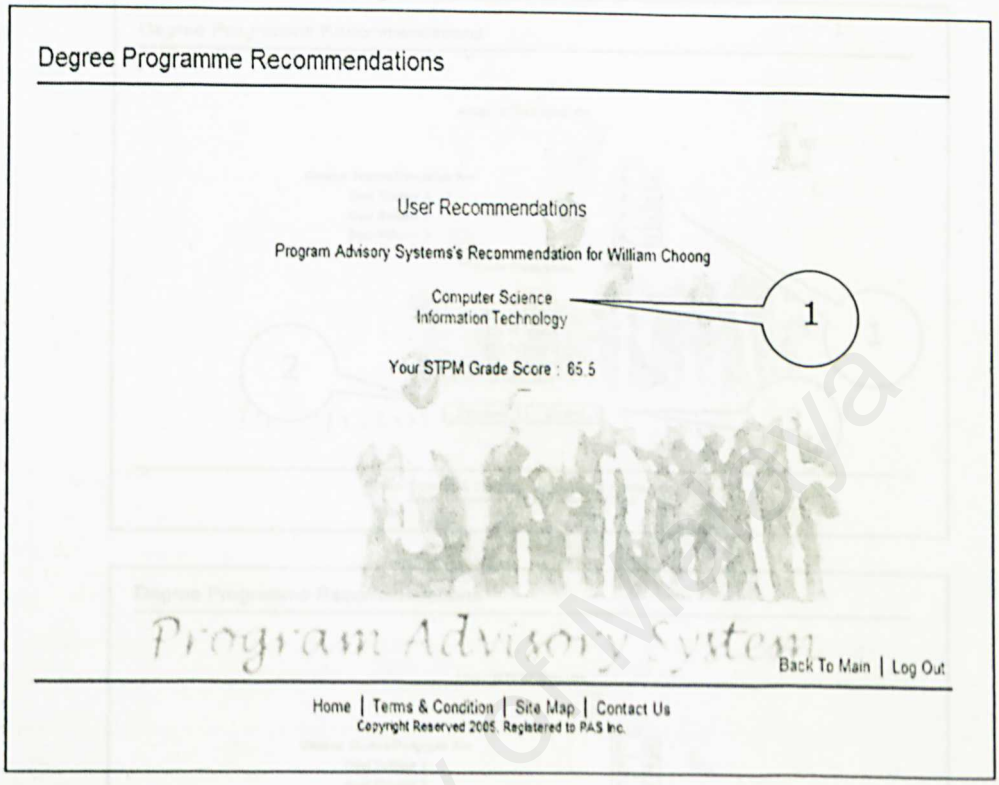
Member's Degree Programme Recommendation Main Screen



Instructions:

1. To view previous recommendations, click on the link here.
2. To proceed to the recommendation process, click on the link here.
3. To proceed back to the Member's Main screen, click on the link 'Back To Main'.

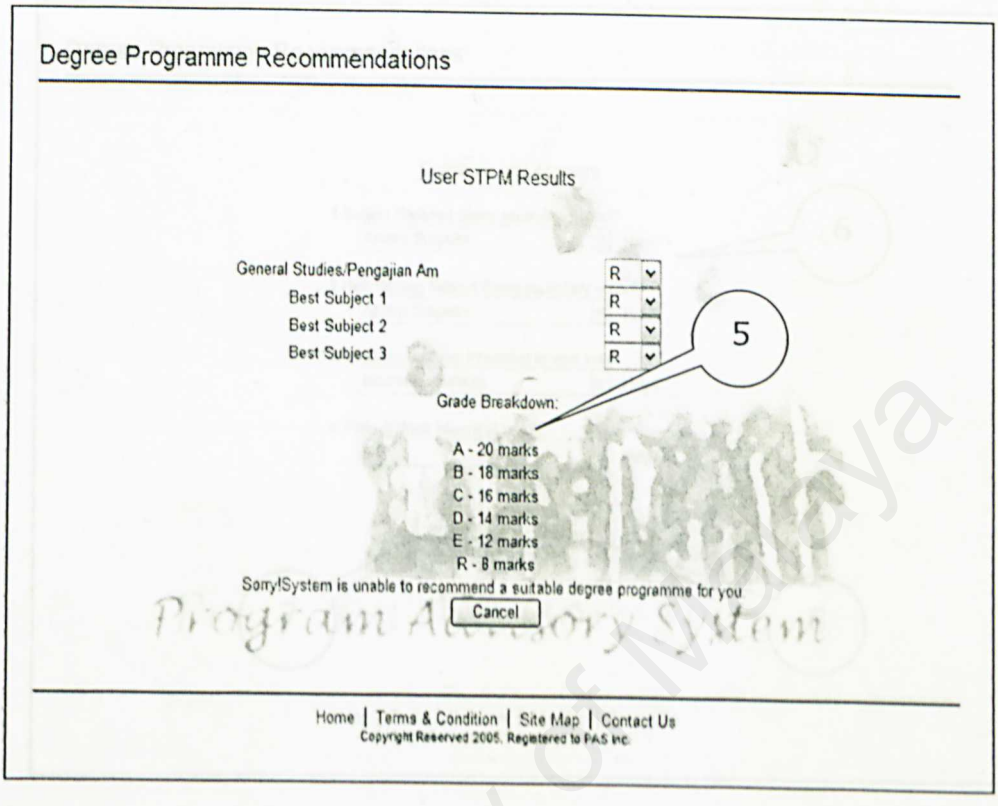
Member's Degree Programme Recommendation Results Screen



Instructions:

1. To view the recommended degree programme, click on the degree programme name here.

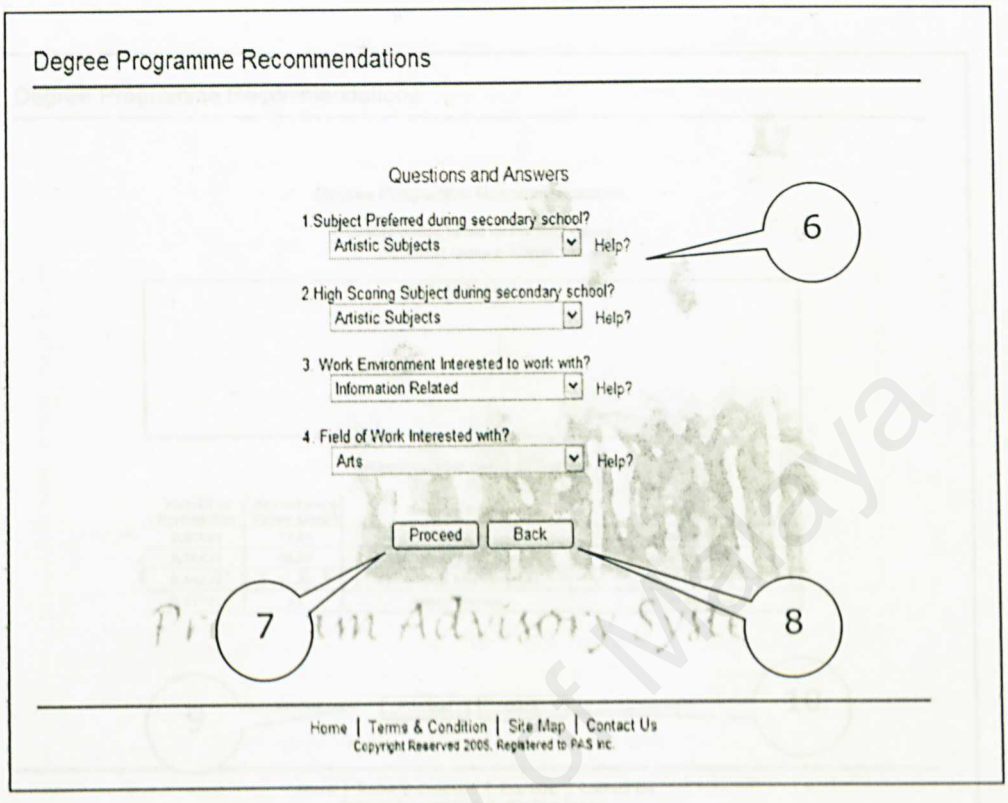
Member's Degree Programme Recommendation Process Screen



Instructions:

5. If your result is way below the minimum accepted value, the system will notify you of the failure of the recommendation evaluation. You will need to click on the Cancel button to redirect you back to you previous screen.

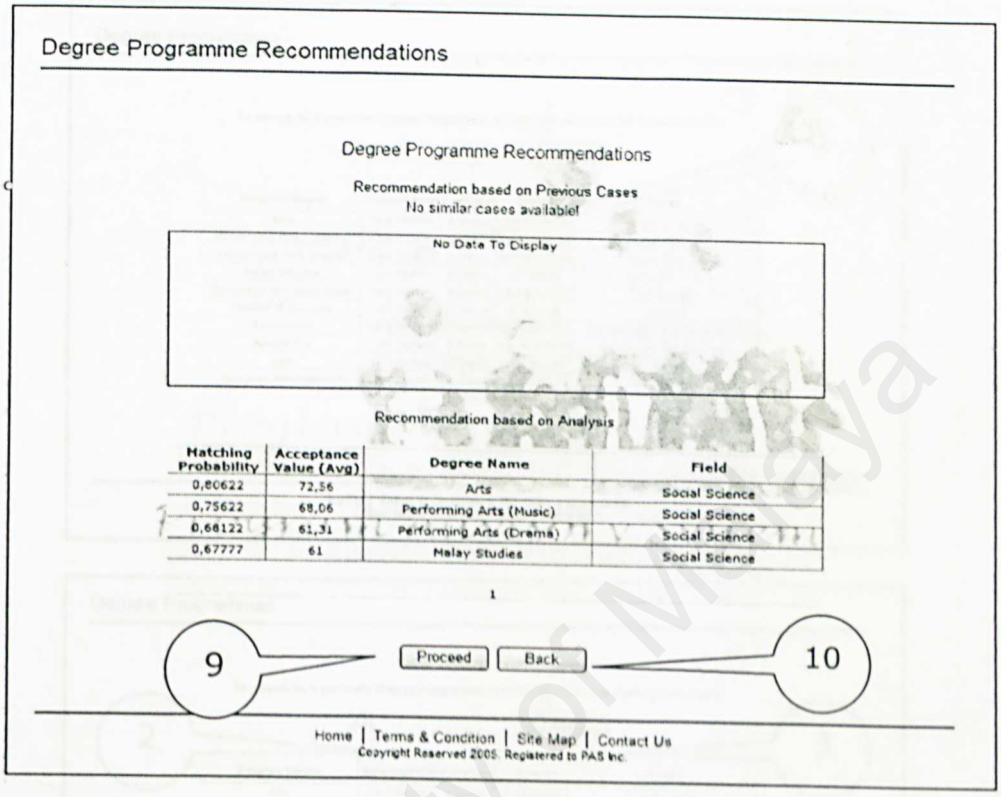
Member's Degree Programme Recommendation Process Screen



Instructions:

6. Fill in the answers to the questions stated. For help, you can click on the 'Help?' link by the side of each question. A brief description of the question and the answer will be given.
7. Click on the Proceed button to carry on with the recommendation evaluation.
8. Click on the Back button to proceed back to the previous screen.

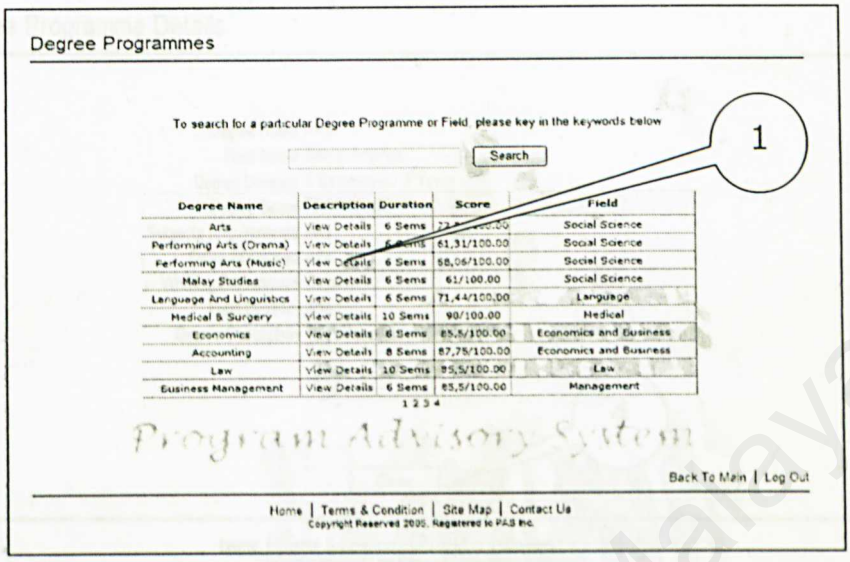
Member's Degree Programme Recommendation Process Screen

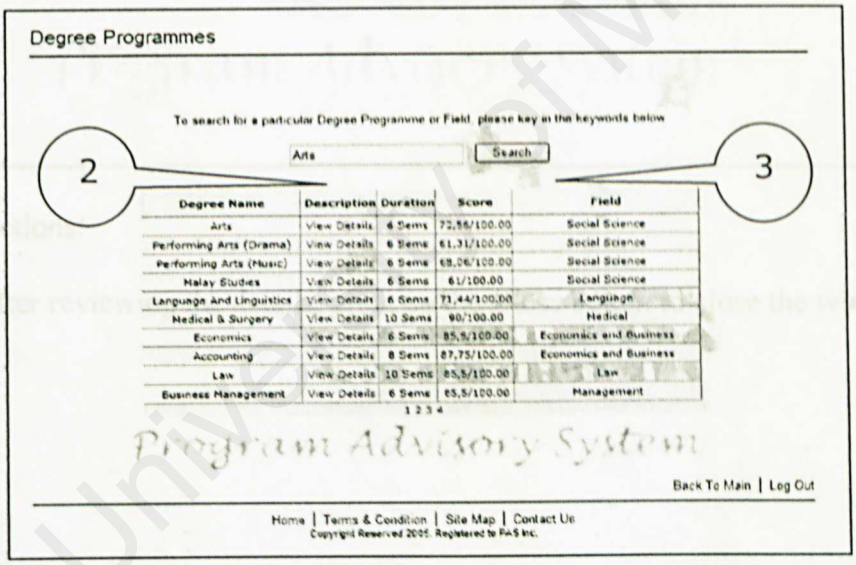


Instructions:

9. Click on the Proceed button to save your top 4 recommendations into the database.
10. Click on the Back button to terminate the recommendation evaluation process and redirect you back to the Recommendation Main screen.

Degree Programme Details Screen

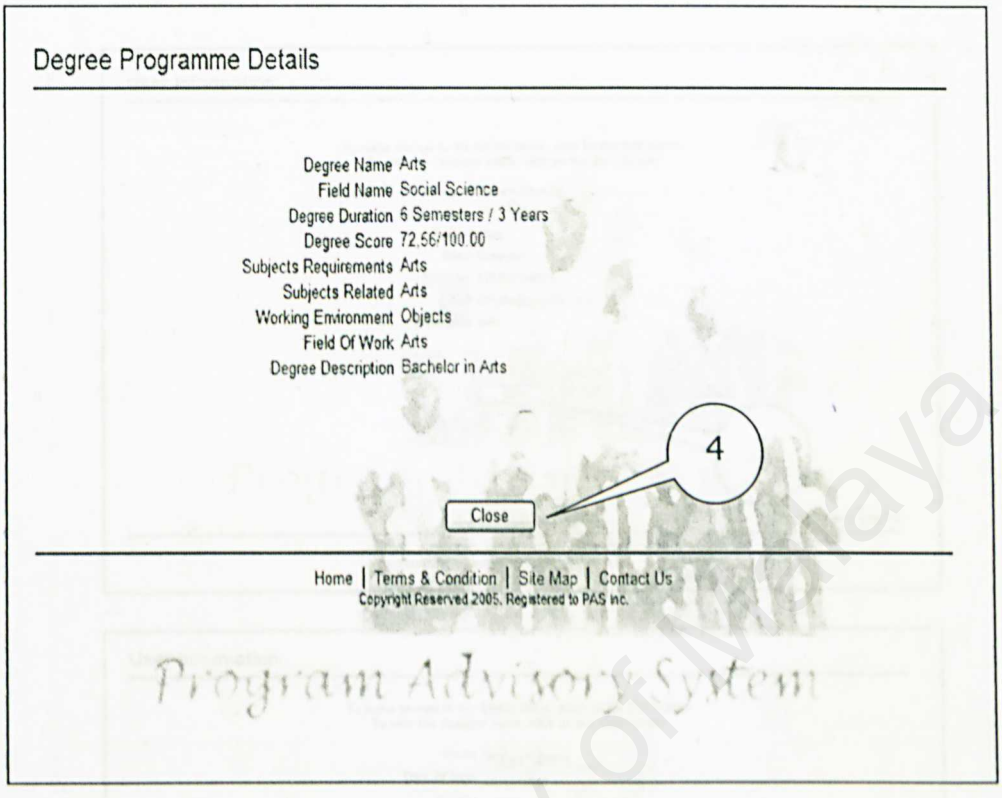




Instructions:

1. To view the details of a particular degree programme, click on the 'View Details' link in the data grid.
2. Type the keyword that you would like to search for in the textbox.
3. Click on the Search button to search for the keyword.

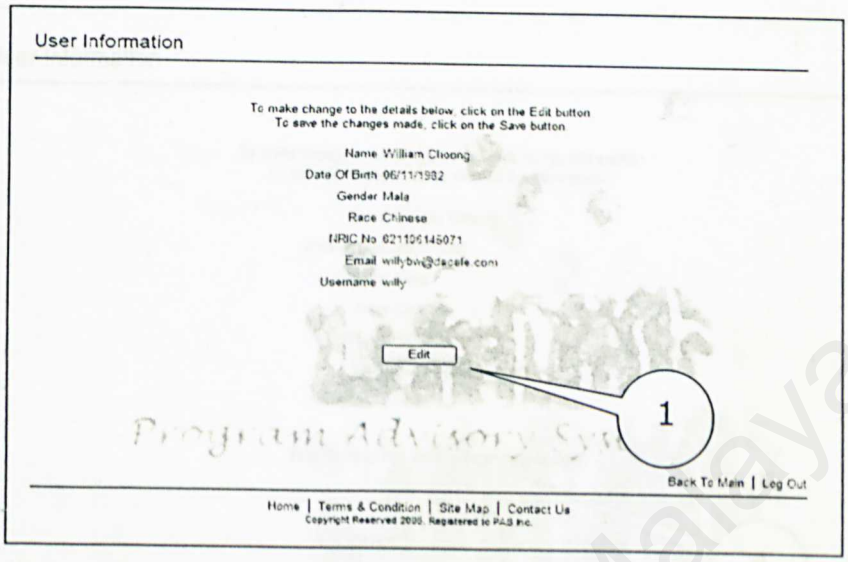
Degree Programme Details Screen

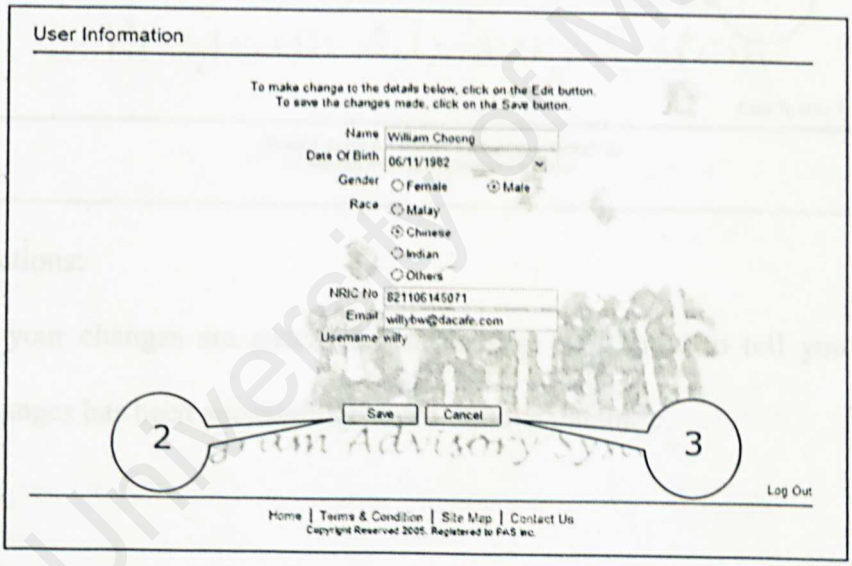


Instructions:

4. After reviewing the details, click on the Close button to close the window.

Member's User Information Screen

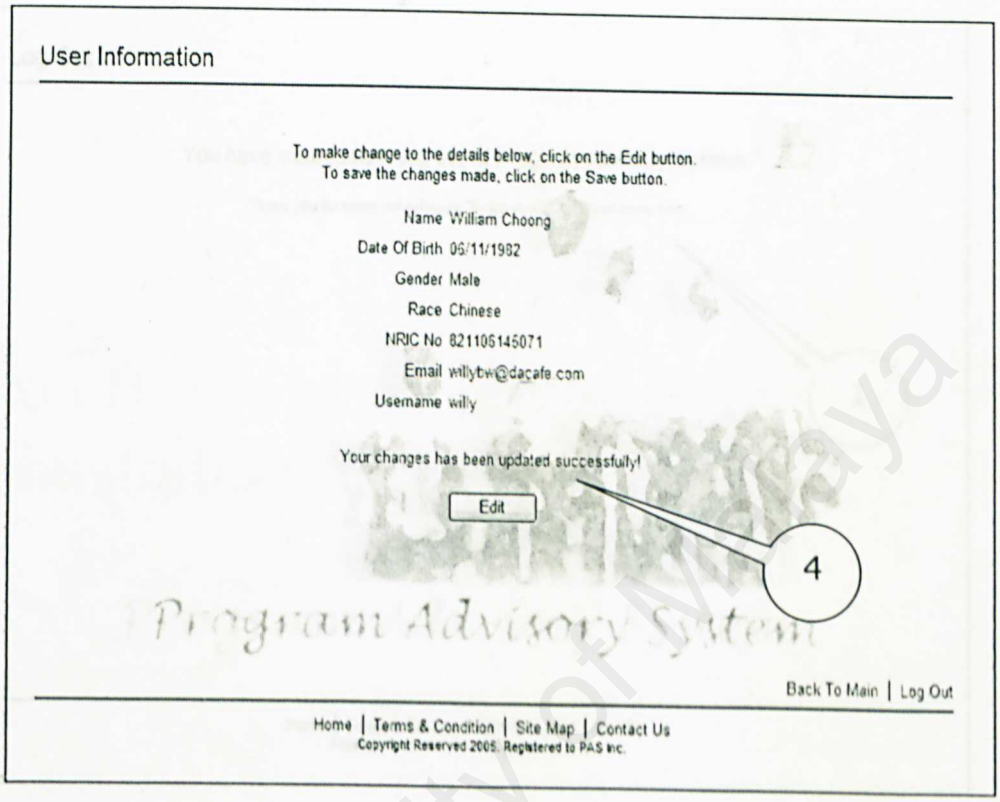




Instructions:

1. Click on the Edit button if you want to make changes on your personal details.
2. Make your desired changes to your particulars. Click on the Save button to make you changes permanent.
3. To cancel your changes, click on the Cancel button.

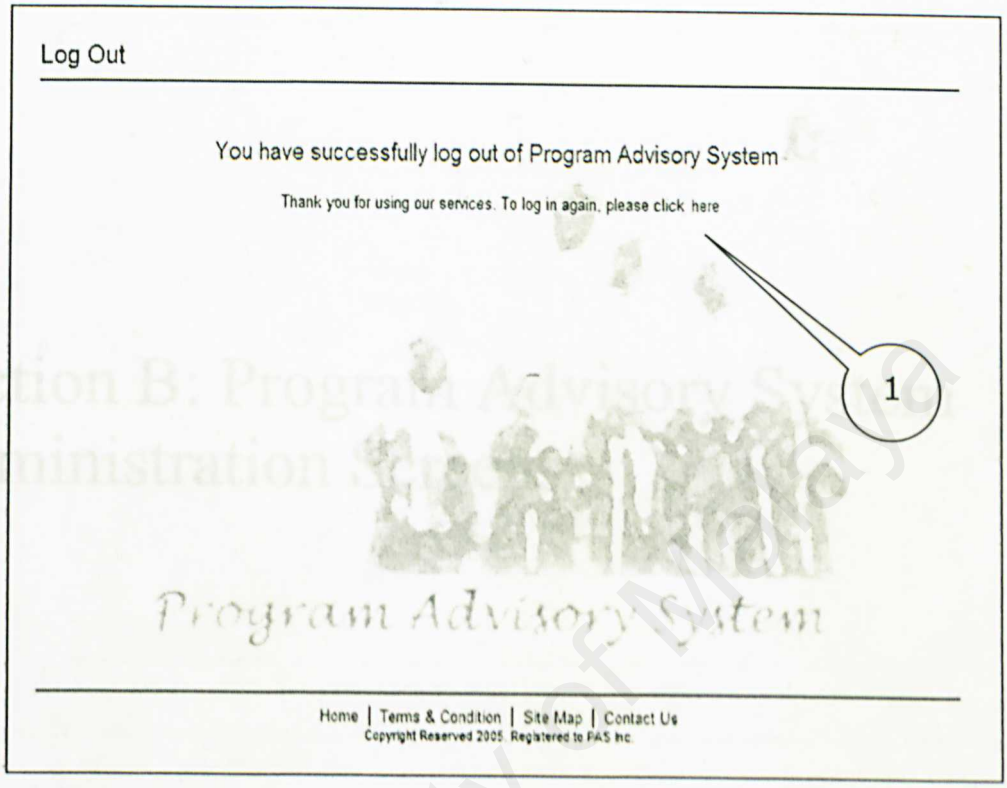
Member's User Information Screen



Instructions:

4. If your changes are successful, a message will show to tell you that your changes has been successfully saved into the database.

Member's Log Out Screen

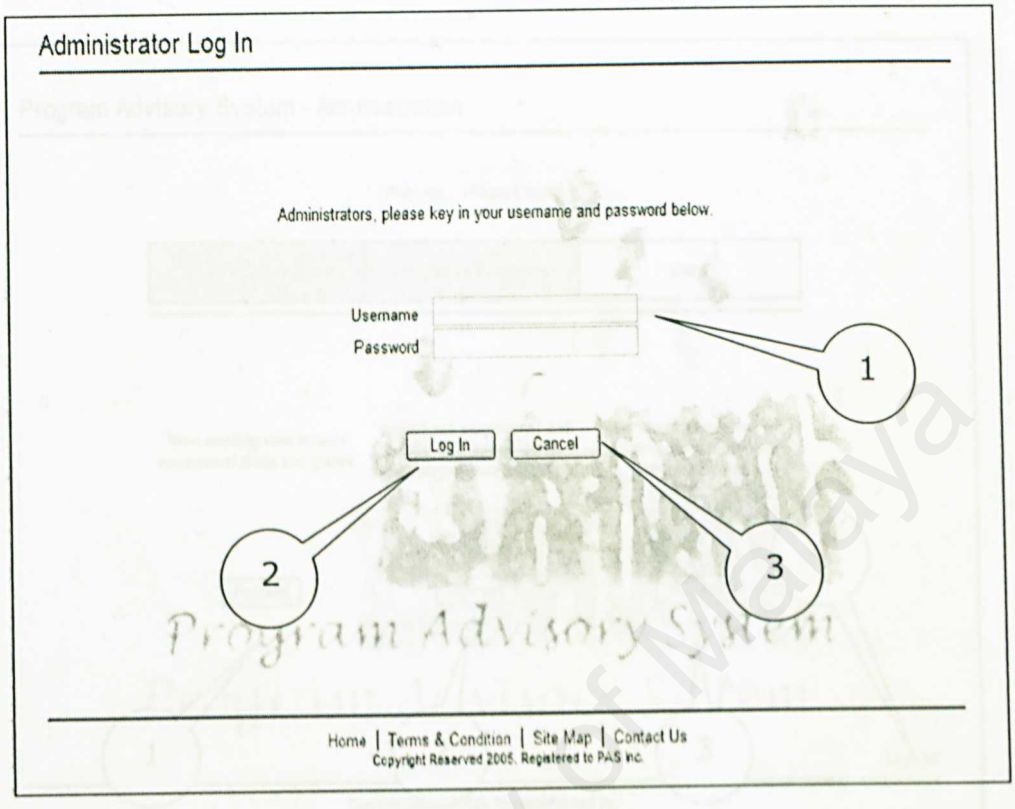


Instructions:

1. If you have click on the Log Out link, you will be directed to the Log Out screen. If you would like to log in back into the system, click on the link here to redirect you back to the Log In screen.

Section B: Program Advisory System Administration Screens

Administrator Log In Screen

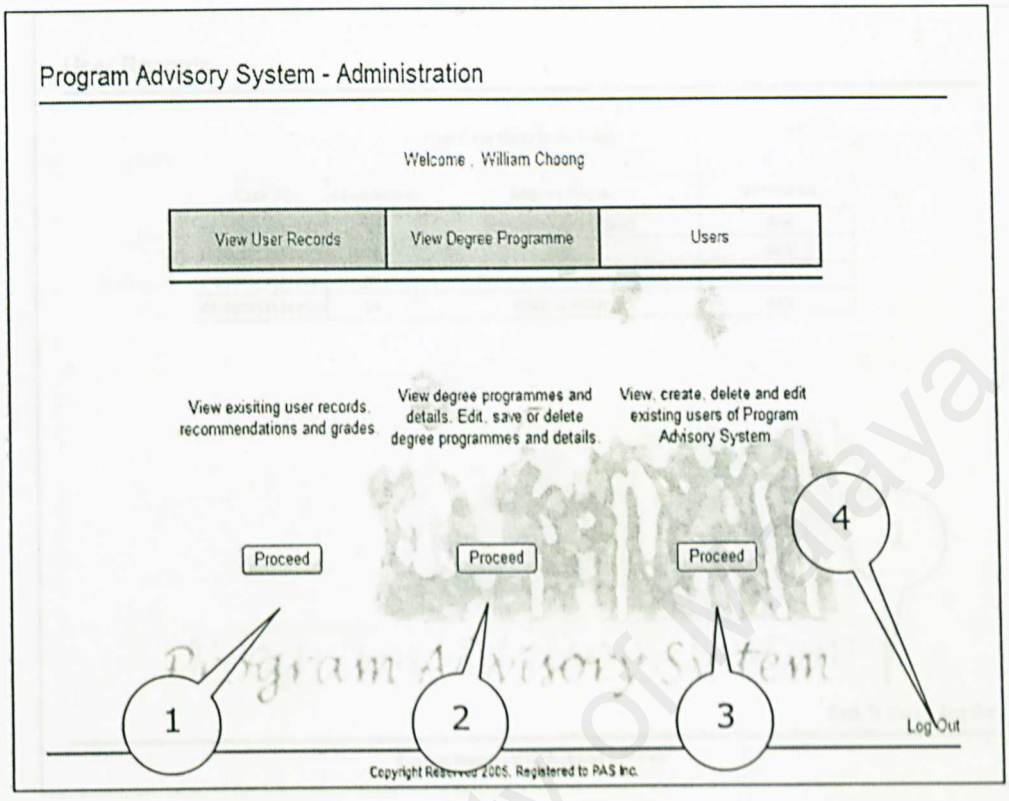


Instructions:

1. Fill in your username and password into the respective textboxes.
2. To proceed to log in using the username and password you filled into the textboxes, click on the Log In button.
3. Else, to cancel your log in and return back to the Program Advisory System Guest Area, click on the Cancel button.

Note: The Administrator Screens are meant for system administrator and system owners to maintain the system. Unauthorized individuals are not allowed to enter the Administrator's screens.

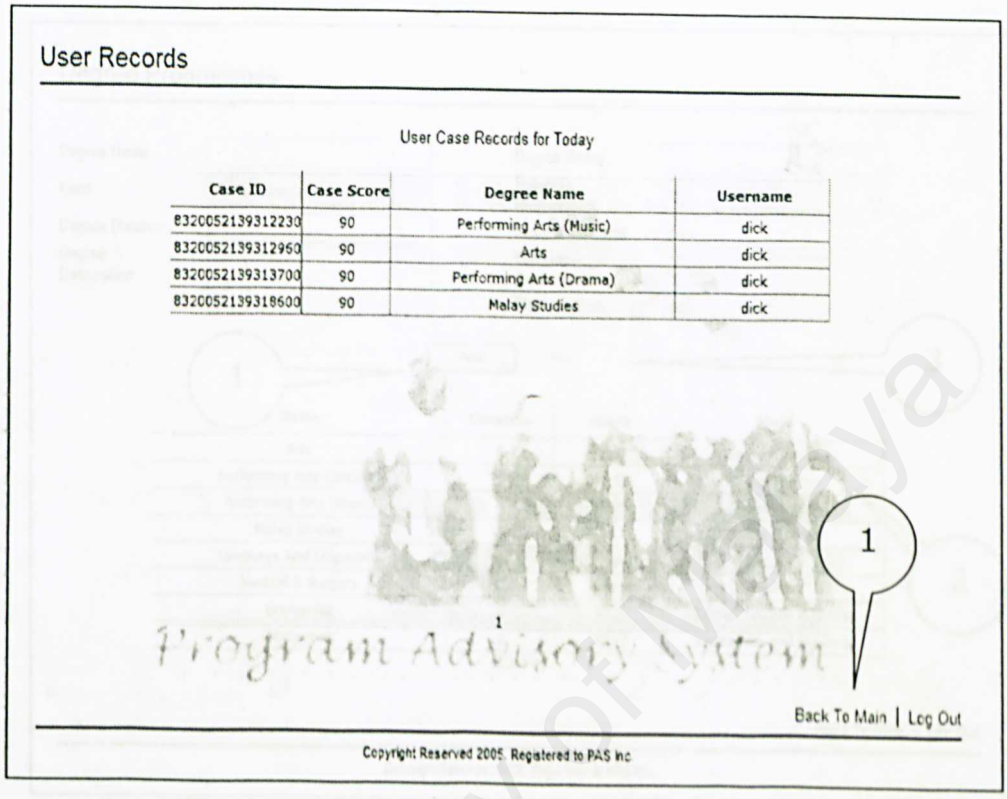
Administrator's Main Screen



Instructions:

1. To view the latest user records, click the first Proceed button to proceed to the View User Records section.
2. To view degree programmes, add new degrees and to edit existing one, click on the second Proceed button to proceed into the View Degree Programme section.
3. To view existing users in the system, click on the third Proceed button to proceed to the Users section.
4. To log out of the Administrator's screen, click on the Log Out link.

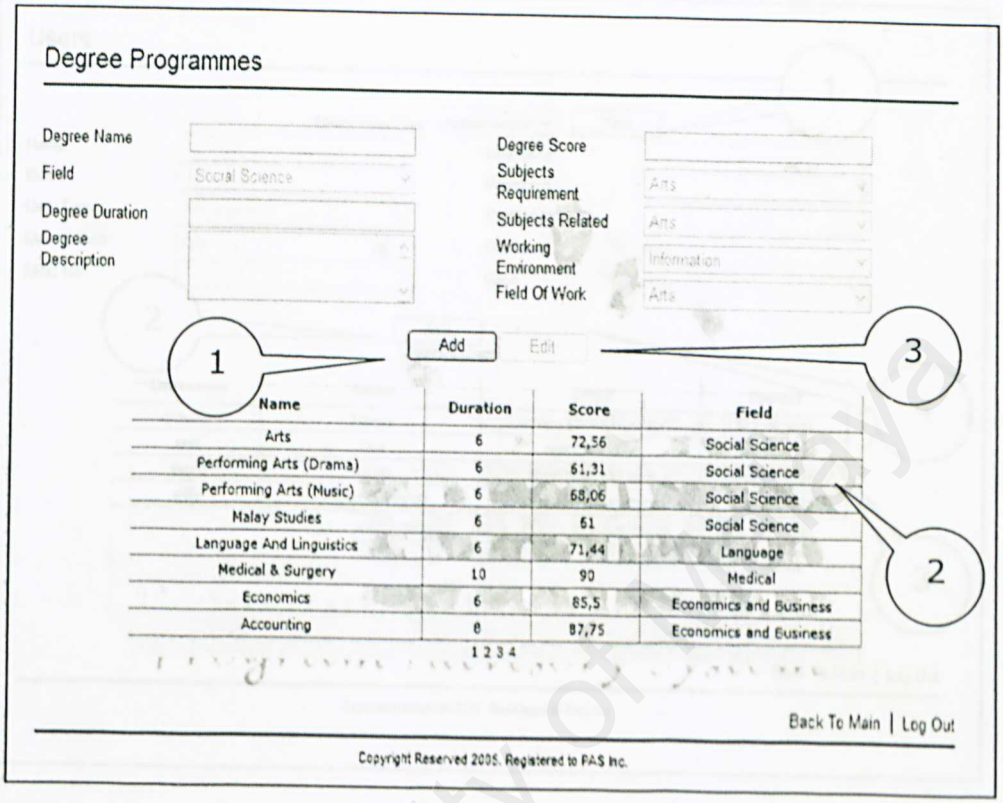
Administrator's User Records Screen



Instructions:

1. To go back to the Administrator's Main screen, click on the Back To Main link.

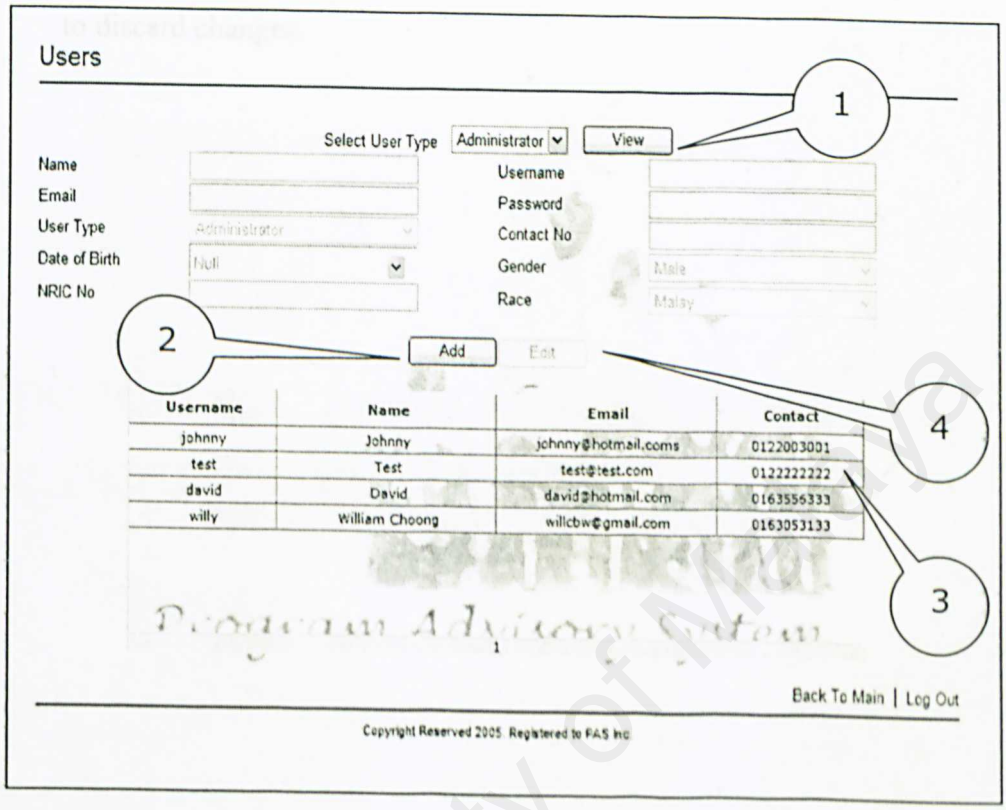
Administrator's Degree Programme Screen



Instructions:

1. To add a new degree programme, click on the Add button. Proceed next to filling in all the required details and click the Save button to save the degree details into the database. Else, click on the Cancel button to discard changes.
2. To edit an existing degree programme, first click on the specific records to change on the data grid.
3. Then when the Edit button is enabled, click on the Edit button. Follow similar steps like in 1 to save or to cancel changes.

Administrator's User Screen



Instructions:

1. To view user types, select the user type in the drop down list and click on the View button refresh the page. After the refresh, the data grid below will display the specific user type selected.
2. To add a new user, click on the Add button. Proceed to filling in all the required details of the user. Then click on the Save button to finalize your input. Else, click on the Cancel button to discard changes.
3. To edit an existing user, first select the type of user to edit. Repeat step 1. Then click on the data grid containing the user record that you would like to change.

4. When the Edit button is enabled, click on it to edit the record. Save the changes made by clicking on the Save button. Else, click on the Cancel button to discard changes.

Appendix B : List of Degree Programmes and Duration

Stream	Field	Program(s)	Years
Arts	Social Science	Arts Performing Arts (Music) Performing Arts (Drama) Malay Studies	3
	Language	Language and Linguistics	3
	Economics and Business	Economics Accounting	3 4
	Islamic Studies	Islamic Studies Usuluddin Syariah	3
	Law	Law	4
	Management	Business Administration	3
	Education	Counselling Children Pre Studies	4
Science	Computing and Information Technology	Computer Science	3
	Dentistry	Dentistry	5
	Pharmacy	Pharmacy	4
	Architectural	Architectural Architectural Science Land Surveyors Quantity Surveyors Real Estate Management	2 3 3 3 3
	Engineering	Civil Engineering Electrical Engineering Chemical Engineering Mechanical Engineering Environmental Engineering CAD/CAM Engineering Product Engineering Production Engineering Bio Production Engineering	4
	Medical	Medical and Surgery Bio Medical Science	5 3
	Science	Science Science in Education	3 4
Arts and Science	Computing and Information Technology	Information Technology	3
	Science	Sport Science	3
	Education	Teaching English as A Second Language	4

Appendix C : List of Grade Scores

Grade Combination	Score
AAAA	90.00%
AAAB	87.75%
AAAC	85.50%
AABB	85.50%
AAAD	83.25%
AABC	83.25%
ABBB	83.25%
AAAE	81.00%
AABD	81.00%
AACC	81.00%
ABBC	81.00%
BBBB	81.00%
AACD	78.75%
AABE	78.75%
ABBD	78.75%
ABCC	78.75%
BBBC	78.75%
AAAR	76.50%
AACE	76.50%
AADD	76.50%
ABBE	76.50%
ABCD	76.50%
ACCC	76.50%
BBBD	76.50%
BBCC	76.50%
AABR	74.25%
AADE	74.25%
ADCE	74.25%
ABDD	74.25%
ACCD	74.25%
BBBE	74.25%
BBCD	74.25%
BCCR	74.25%
AACR	72.00%
AAEE	72.00%
ABBR	72.00%

Grade Combination	Score
ABDE	72.00%
ACCE	72.00%
ACDD	72.00%
BBCE	72.00%
BBEE	72.00%
BCDD	72.00%
CCCC	72.00%
AADR	69.75%
ABDR	69.72%
ABEE	69.75%
ACDE	69.75%
ADDD	69.75%
BBBR	69.75%
BBER	69.75%
BCDE	69.75%
BDDD	69.75%
CCCD	69.75%
AAER	67.50%
ABDR	67.50%
ACCR	67.50%
ACEE	67.50%
ADDE	67.50%
BBCR	67.50%
BCCD	67.50%
BDDE	67.50%
BDDD	67.50%
CCCE	67.50%
CCDD	67.50%
ABER	65.25%
ACDR	65.25%
ADEE	65.25%
BCCC	65.25%
BCER	65.25%
BDEE	65.25%
BDDE	65.25%
CCDE	65.25%

Grade Combination	Score
CDDD	65.25%
ACER	63.00%
ADDR	63.00%
AEEE	63.00%
BCCE	63.00%
BDDR	63.00%
BDEE	63.00%
CCCR	63.00%
CCEE	63.00%
CDDE	63.00%
DDDD	63.00%
ADER	60.75%
BDER	60.75%
BDDR	60.75%
BEEE	60.75%
CCDR	60.75%
CDEE	60.75%
DDDE	60.75%
AEER	58.50%
BDER	58.50%
CCER	58.50%
CDDR	58.50%
CEEE	58.50%
DDEE	58.50%
BEER	56.25%
CDER	56.25%
DDDR	56.25%
DEEE	56.25%
CEER	54.00%
DDER	54.00%
EEEE	54.00%
DEER	51.75%
EEER	49.50%
RRRR	36.00%

Appendix D : List of Degree Programmes with Requirements and Scores

Field	Degree Programme	Duration	Requirement 1 & 2	Requirement 3	Requirement 4	Min Score	Max Score	Average
Social Science	Arts	3	Art	Objects	Art	55.125	90	72.5625
	Performing Arts(Music)	3	Art	People	Art	48.375	74.25	61.3125
	Performing Arts(Drama)	3	Art	People	Art	48.375	87.75	68.0625
	Malay Studies	3	Art	Information	Teaching,Education,Research	50	72	61
Language	Language and Linguistics	3	Language	Information	Teaching,Education,Research / Publishing & Media	57.37	85.5	71.435
Economics and Business	Economics	3	Calculation	Information	Banking & Finance	81	90	85.5
	Accounting	4	Calculation	Information	Banking & Finance	85.5	90	87.75
Islamic Studies	Islamic Studies	3		Information	Teaching,Education,Research			
	Usuluddin	3		Information	Teaching,Education,Research			
	Syariah	3		Information	Teaching,Education,Research			
Law	Law	4	Language	Words	Law	81	90	85.5
Management	Business Administration	3	Calculation	Information	Banking & Finance	81	90	85.5
Education	Counselling	4	Technology	Information	Social & Healthcare	63	85.5	74.25
	Children Pre Studies	4	Technology	Information	Social & Healthcare	47.25	81	64.125

Field	Degree Programme	Duration	Requirement 1 & 2	Requirement 3	Requirement 4	Min Score	Max Score	Average
Computing and Information Technology	Computer Science	3	Technology	Information	Computing	72	90	81
Dentistry	Dentistry	5	Technology	People	Social & Healthcare	85.8	90	87.9
Pharmacy	Pharmacy	4	Technology	People	Social & Healthcare	83.25	90	86.625
Architecture	Architectural	2	Technology	Objects	Construction			
	Architectural Science	3	Technology	Objects	Construction	81	90	85.5
	Land Surveyors	3	Technology	Objects	Construction	69.75	81	75.375
	Quantitry Surveyors	3	Technology	Objects	Construction	75.75	90	82.875
	Real Estate Management	3	Technology	Objects	Construction	67.5	91	79.25
Engineering	Civil Engineering	4	Technology	Objects	Engineering	76.5	87.75	82.125
	Electrical Engineering	4	Technology	Objects	Engineering	87.75	90	88.875
	Chemical Engineering	4	Technology	Objects	Engineering	83.25	90	86.625
	Mechanical Engineering	4	Technology	Objects	Engineering	85.5	90	87.75
	CAD/CAM Engineering	4	Technology	Objects	Engineering	83.25	90	86.625
	Environmental Engineering	4	Technology	Objects	Engineering	69.75	85.5	77.625
	Product Engineering	4	Technology	Objects	Engineering	78.75	87.75	83.25
	Production Engineering	4	Technology	Objects	Engineering	78.75	87.75	83.25
	Bio Production Engineering	4	Technology	Objects	Engineering			
Medical	Medical and Surgery	5	Technology	People	Social & Healthcare	90	90	90
	Bio Medical Science	3	Technology	Information/People	Social & Healthcare	83.25	90	86.625
Science	Science	3	Technology	Information	Research	46.125	90	68.0625
	Science in Education	4	Technology	People	Teaching,Education,Research			
Computing and Information Technology	Information Technology	3	Technology/Information	Information	Computing	76.5	90	83.25
Science	Sports Science	3	Technology	People	Social & Healthcare	40.5	65.25	52.875
Education	Teaching English As A Second Language	4	Language	People	Teaching,Education,Research	55.13	78.75	66.94